

Optimization for Image Segmentation

by

Meng Tang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2019

© Meng Tang 2019

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: Dr. M. Pawan Kumar
Associate Professor, Department of Engineering Science
University of Oxford

Supervisor(s): Dr. Yuri Boykov
Professor, School of Computer Science
University of Waterloo

Internal Member: Dr. Yaoliang Yu
Assistant Professor, School of Computer Science
University of Waterloo

Dr. Pascal Poupart
Professor, School of Computer Science
University of Waterloo

Internal-External Member: Dr. Paul Fieguth
Professor, Department of Systems Design Engineering
University of Waterloo

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

I hereby declare that I am the sole author of Chapter 1 on introduction and motivation.

Chapter 2 is based on published material [217] co-authored with Ismail Ben Ayed and Yuri Boykov. I am the first author and main contributor of the paper [217]. I conducted all the experiments.

Chapter 3 is mostly based on published material [223] co-authored with Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. All authors contribute equally to the paper [223]. Chapter 3 is partially based on related published material [218, 222, 158]. I am the first author and main contributor of the paper [218, 222]. Dmitrii Marin is the first author and main contributor of the co-authored paper [158].

Chapter 4 is mostly based on published material [219, 224] for which I am the first author and main contributor. The work [219] is done when I was an intern at Disney Research Zurich, supervised by Federico Perazzi, Abdelaziz Djelouah, and Christopher Schroers. I proposed the main idea and conducted all the experiments for the paper [219, 224]. Chapter 4 is partially based on published material [157, 115] for which I am a co-author and a minor contributor.

I hereby declare that I am the sole author of Chapter 5 on conclusion and future work.

Abstract

Image segmentation, i.e., assigning each pixel a discrete label, is an essential task in computer vision with lots of applications. Major techniques for segmentation include for example Markov Random Field (MRF), Kernel Clustering (KC), and nowadays popular Convolutional Neural Networks (CNN). In this work, we focus on *optimization* for image segmentation. Techniques like MRF, KC, and CNN optimize MRF energies, KC criteria, or CNN losses respectively, and their corresponding optimization is very different. We are interested in the synergy and the complementary benefits of MRF, KC, and CNN for interactive segmentation and semantic segmentation. Our first contribution is *pseudo-bound optimization* for binary MRF energies that are high-order or non-submodular. Secondly, we propose *Kernel Cut*, a novel formulation for segmentation, which combines MRF regularization with Kernel Clustering. We show why to combine KC with MRF and how to optimize the joint objective. In the third part, we discuss how deep CNN segmentation can benefit from non-deep (i.e., shallow) methods like MRF and KC. In particular, we propose *regularized losses* for weakly-supervised CNN segmentation, in which we can integrate MRF energy or KC criteria as part of the losses. Minimization of regularized losses is a principled approach to semi-supervised learning, in general. Our regularized loss method is very simple and allows different kinds of regularization losses for CNN segmentation. We also study the optimization of regularized losses beyond gradient descent. Our regularized losses approach achieves state-of-the-art accuracy in semantic segmentation with near full-supervision quality.

Acknowledgements

Firstly, I would like to thank my supervisor Prof. Yuri Boykov who introduced computer vision research to me, and I got obsessed ever since. I feel fortunate to work with him in the last few years. I am very grateful for him spending a sheer amount of time meeting and discussing with me. Yuri generously shared with me his keen passion for research, which kept me motivated. He gave helpful and insightful advice whenever I got distracted or lost in research directions. He taught me to be a pure researcher driven by curiosity and clarity. One valuable advice I learned from Yuri is to be informed of what is out there in research, but do not blindly follow "hot" topics. Also, I can not agree more on Yuri's research philosophy towards simplicity.

Secondly, I would like to thank the examination committee including Prof. M. Pawan Kumar, Prof. Yaoliang Yu, Prof. Pascal Poupart, and Prof. Paul Fieguth. Their feedback, comments, and criticism are critical to improving this work. Special thanks to my external examiner Prof. M. Pawan Kumar who provided very detailed comments.

I take this opportunity to thank my informal co-supervisor, Prof. Ismail Ben Ayed (ETS Montreal), with whom I had fruitful collaboration since 2014. I enjoyed the many discussions we had on clustering. I still remember the long discussion the night before Ismail's second child, Adam, was born. Ismail is someone I can talk to if I have a random thought and immature research idea. I thank members of ETS Montreal vision group, including Imtiaz Ziko, Jose Dolz, Prof. Christian Desrosiers, and Prof. Herve Lombaert for their interest and comment on my work.

I would like to thank Prof. Olga Veksler with whom I co-authored my very first paper in computer vision. Since then, Olga has been helping me in many different ways even beyond work when I encountered some difficulties in life.

I thank my lab mates Dr. Lena Gorelick, Dr. Hossam Isack, Dmitrii Marin, Egor Chesakov, Richard Zhang, Freddy Liu, and Towaki Takikawa. Thank you for all the discussions! Without you, my life will be bored. Special thank to Dmitrii Marin with whom I work closely and share office for a few years.

I thank the people I worked with in industry during internships. I would like to thank my mentors Edward Hsiao (now at Waymo) and Douglas Gray when I was an intern in Amazon A9's visual search group. I thank Abdelaziz Djelouah, Christopher Schroers, and Federico Perazzi (now at Adobe) for taking me as an intern in Disney Research Zurich. I also thank Prof. Pascal Poupart for hosting my internship at BorealisAI. I would like to thank Prof. Ramin Zabih and Dr. Chen Wang in Google for their interest in my work.

Lastly but most importantly, I thank my family for their love and support these years.

Dedication

Dedicated to my mother Xia Wu, my father Haiqing Tang, and my sister Ye Tang.

Dedicated to my wife Dana Wang who kindly agreed to appear in some figures of this thesis for illustration purpose.

Table of Contents

List of Tables	xii
List of Figures	xvi
1 Introduction	1
1.1 Image Segmentation	1
1.2 Notation and Conventions	3
1.3 Overview of Segmentation Techniques	4
1.3.1 A Toy Example: K-means	6
1.3.2 Kernel Clustering (KC)	7
1.3.3 Markov Random Field (MRF)	10
1.3.4 Convolutional Neural Networks (CNN)	18
1.4 Our Motivation and Contribution	20
1.4.1 Bound Optimization	20
1.4.2 Weakly-supervised Segmentation and Semi-supervised Learning	23
1.4.3 Combining KC, MRF, and CNN	25
1.5 Outline of the Thesis and Publication	27
2 Pseudo-bound Optimization for Binary MRF Energies	29
2.1 Introduction	29
2.1.1 Bound optimization	30

2.1.2	Motivation and Contributions	30
2.2	Parametric Pseudo-Bound Cuts (pPBC)	33
2.2.1	Our pseudo-bound framework	33
2.2.2	Overview of <i>parametric maxflow</i>	35
2.3	Examples of pseudo-bounds	35
2.3.1	High-order energies	36
2.3.2	Non-submodular pairwise energies	39
2.4	Experiments	40
2.4.1	Appearance entropy based segmentation	40
2.4.2	Matching color distributions	43
2.4.3	Curvature Regularization	43
2.4.4	Deconvolution	44
2.5	Conclusion	45
3	Kernel Clustering Meets MRF Regularization	46
3.1	Introduction: Terminology and Motivation	46
3.1.1	Notation	48
3.1.2	Our approach summary	49
3.1.3	Motivation and Related work	51
3.1.4	Main contributions	53
3.2	Background on Regularization and Clustering	55
3.2.1	Overview of MRF regularization	55
3.2.2	Overview of K-means and clustering	56
3.2.3	NC objective and its relation to AA, AC, and <i>kKM</i>	67
3.2.4	Optimization methods for kernel clustering	69
3.3	Kernel Bounds	71
3.3.1	Bound optimization and K-means	71
3.3.2	Kernel Bounds for AA, AC, and NC	72

3.3.3	Move-making algorithms	76
3.4	Data Embeddings and Spectral Bounds	78
3.4.1	Exact and approximate embeddings ϕ for k KM	78
3.4.2	Spectral Bounds for AA, AC, and NC	83
3.4.3	Relation to spectral clustering	88
3.5	Experiments	92
3.5.1	MRF helps Kernel & Spectral Clustering	93
3.5.2	Kernel & Spectral Clustering helps MRF	96
4	Regularized Losses for Weakly Supervised CNN Segmentation	106
4.1	Introduction and Motivation	106
4.1.1	Regularization for Weakly-supervised Segmentation	108
4.1.2	Regularization for Semi-supervised Learning	109
4.1.3	Our Contributions	111
4.2	Related Work	112
4.2.1	CNN for Semantic Segmentation	112
4.2.2	Weakly-supervised CNN Segmentation	114
4.2.3	Regularization for Neural Networks	115
4.3	Our Regularized Losses	116
4.3.1	MRF Energy as Loss	117
4.3.2	Clustering Criterion as Loss	119
4.3.3	Linear Constraint as Loss	120
4.4	Beyond Gradient Descent for Regularized Losses	121
4.4.1	Alternative Direction Method (ADM)	122
4.5	Experiments	124
4.5.1	Visualization of Gradients	125
4.5.2	Comparison of Regularized Losses	125
4.5.3	CRF as Loss, Post-processing or Trainable Layers	127
4.5.4	GD vs ADM for Grid CRF Loss	130
4.5.5	Fully Supervision and Semi-supervision	134

5 Conclusion and Future Work	135
References	138
APPENDICES	162
A Matlab Code for Kernel and Spectral Bound	163
A.1 Kernel Bound	163
A.2 Spectral Bound	164
B PyTorch Code for Regularized Loss	167
B.1 DenseCRF Loss	167

List of Tables

1.1	Frequently used notations for segmentation in this thesis.	3
2.1	Auxiliary functions [90] and weighted bound relaxation term for pPBC-T. .	40
2.2	A table beside a figure	42
2.3	Statistics of pPBC and GrabCut [200] over the GrabCut database.	42
2.4	Matching color distribution (KL or Bhattacharyya distance) with Auxiliary Cuts [18], FTR [91], pPBC and its limited version with $\lambda \leq 0$ for the pseudo-bounds. Mean energy and error are reported.	43
2.5	Average energy with 10 random noisy images.	45
3.1	Our notation for segmentation of points $p \in \Omega$ uses discrete labels and binary indicators, see Tab. 1.1. Without much ambiguity, segment S^k could mean both a subset of Ω or its indicator vector, <i>i.e.</i> S^k is either an element of <i>power set</i> $\mathcal{P}(\Omega)$ or a vector $\{0, 1\}^{ \Omega }$. While unnecessary for most of the technical results in this paper, in the context of <i>relaxation</i> methods it is easy to switch to an alternative representation (the last column) where segment S^k becomes a relaxed vector $[0, 1]^{ \Omega }$. This is consistent with a common (relaxed) <i>assignment matrix</i> representation of segmentation S where integer label S_p becomes a vector on probability simplex Δ^K specifying pixel's support/distribution over K labels.	49

3.2	Examples of “ <i>graph cut</i> ” criteria appearing in the contexts of (MRF) <i>regularization</i> and <i>kernel clustering</i> that can be used in joint energy (3.1) simultaneously. The cut cost, <i>i.e.</i> the sum of edge weights w_{pq} or affinities A_{pq} between the segments, can be represented via matrices $\mathcal{W} = [w_{pq}]$ or $A = [A_{pq}]$, and segment indicators S^k , see Tab. 3.1. The right column differs only by normalization over segment <i>cardinality</i> $ S^k $ or <i>weighted cardinality</i> $d'S^k$ where $d := A\mathbf{1}$ are <i>node degrees</i>	51
3.3	<i>K-means and related clustering criteria</i> : Basic K-means (A) minimizes clusters variances. It works as Gaussian model fitting. Fitting more complex models like elliptic Gaussians [215, 201, 60], exponential distributions [8], GMM or histograms [263, 200] corresponds to <i>probabilistic K-means</i> [114] in (B). Kernel clustering via <i>kernel K-means</i> (C) using more complex data representation.	58
3.4	<i>Kernel bounds</i> for different kernel clustering objectives $E_A(S)$ can be derived from (3.49) and (3.52) using corresponding \mathcal{K} and w . The second column shows formulations of these objectives $E_A(S) \equiv \sum_k e(S^k)$ using functions e over segment indicator vectors $S^k \in \{0, 1\}^{ \Omega }$. (3.52) gives a unary (linear) upper bound for $E_A(S)$ at S_t based on the first-order Taylor approximation of <i>concave relaxation</i> function $\hat{e}: \mathcal{R}^\Omega \rightarrow \mathcal{R}^1$ (3.48).	74
3.5	<i>Spectral bounds</i> for objectives $E_A(S)$. The third column shows p.d. kernel matrices \mathcal{K} for the equivalent <i>kKM</i> energy (3.21). Eigen decomposition for matrices in the forth column defines our Euclidean embedding $\tilde{\phi}_p \in \mathcal{R}^m$ (fifth column) isometric to \mathcal{K} (3.59). Thus, K-means over $\tilde{\phi}_p$ approximates <i>kKM</i> (3.21). Bounds for KM (last column) follow from Th. 1 where $\mu_t = \{\mu_{S^k}\}$ are means (3.16) and $\mu_t^w = \{\mu_{S^k}^w\}$ are weighted means (3.32). Functions $F(S, m)$ and $F^w(S, m)$ are modular (linear) w.r.t. S , see (3.46, 3.47).	85

3.6	<i>Spectral relaxation</i> and discretization heuristics for objectives for kernel clustering objectives $E_A(S)$ for affinity A . The corresponding <i>degree</i> matrix D is diagonal with elements $d_p := \sum_q A_{pq}$. To extract integer labeling from the relaxed solutions produced by the eigen systems (second column), spectral methods often apply basic KM to some ad hoc data embedding ϕ (last column) based on the first K unit eigenvectors \mathbf{u} , the rows of matrix U^K . While our main text discusses some variants, the most basic idea [211, 235] is to use the columns of U^K as embedding $\tilde{\phi}_p$. For easier comparison, the last column also shows equivalent representations of this embedding based on the same eigen decompositions $V'AV$ as those used for our isometry eigenmaps in Tab. 3.5. In contrast, our embeddings are derived from justified approximations of the original non-relaxed AA , AC , or NC objectives. Note that NC corresponds to a <i>weighted</i> case of K-means with data point weights $w_p = d_p$ [9, 64], see (3.41) in Sec. 3.2.3.	91
3.7	Results of spectral clustering (K-means on eigenvectors) and our Kernel Cut & Spectral Cuts on BSDS500 dataset. For this experiment mPb-based kernel is used [6].	94
3.8	Box-based interactive segmentation (Fig. 3.19). Error rates (%) are averaged over 50 images in GrabCut dataset. KernelCut-Gau-NC means KernelCut for fixed width Gaussian kernel based normalized cut objective. . . .	98
3.9	Seeds-based interactive segmentation (Fig. 3.20). Error rates (%) are averaged over 82 images from Berkeley database. Methods get the same seeds entered by four users. We removed 18 images with multiple nearly-identical objects from 100 image subset in [160]. (GrabCut and KernelCut-KNN-AA give 3.8 and 3.0 errors on the whole database.)	99
4.1	mIOU on PASCAL VOC2012 <i>val</i> set. Our flexible framework allows various types of regularization losses for weakly supervised segmentation, e.g. normalized cut, CRF or their combinations (KernelCut [222]) as joint loss. We achieved the state-of-the-art with scribbles. In () shows the offset to the result with full masks.	126
4.2	Ablation study of CRF as loss, post-processing [45] or trainable layers [260] for weakly supervised segmentation with scribbles for DeepLab-VGG16. . . .	128

4.3	Tag-based weak supervision. We train with different combinations of the losses in SEC [121] and our CRF loss. Replacing the constrain-to-boundary loss in SEC [121] by CRF loss gives minor improvement in accuracy, but training with regularized loss with gradient descent is faster since no iterative CRF inference is needed. We also compare to a variant (*) of SEC without back-propagation of the CRF inference layer. Parenthesis (·) show the time for the constrain-to-boundary loss layer or our direct loss layer.	128
4.4	ADM gives better grid CRF losses than gradient descend (GD). †We randomly selected 1,000 training examples.	130
4.5	Weakly supervised segmentation results for different choices of network architecture, regularized losses and optimization via gradient descent or ADM. We show mIOU on <i>val</i> set of PASCAL 2012. ADM consistently improves over GD for different networks for grid CRF. Our grid CRF with ADM is competitive to previous state-of-the-art dense CRF (with GD) [224].	132
4.6	Pixel-wise accuracy on <i>val</i> set of PASCAL 2012. Top 3 rows: accuracy over all pixels. Middle 3 rows: accuracy for pixels within 16 pixels away from semantic boundaries. Bottom 3 rows: accuracy for pixels within 8 pixels away from semantic boundaries. Pixels closer to boundaries are more likely to be mislabeled. Our ADM scheme improves over GD for grid CRF loss consistently for different networks. Note that weak supervision with our approach is almost as good as full supervision.	132
4.7	Negative effect of regularization loss for full supervision.	134

List of Figures

1.1	Segmentation problems of different kinds [200, 177, 74].	1
1.2	Segmentation Techniques.	4
1.3	K-means segmentation for different number of segments.	6
1.4	Spectral method for normalized cut segmentation.	8
1.5	Factors/Cliques of different orders for 2D image.	11
1.6	GrabCut (Iterated Graph Cut) [200, 30] for interactive segmentation with box.	12
1.7	Typical CRF models used for image segmentation. Left: Sparse or grid CRF [30]; Right: Dense CRF [130].	13
1.8	Architecture of LeNet-5 for digits recognition [138]. ©IEEE	18
1.9	Fully convolutional neural networks for semantic segmentation [150]. ©IEEE	19
1.10	<i>Illustration of the general bound optimization procedure:</i> Iteration t of optimizing function $F(S)$ using auxiliary functions (bounds) $a_t(S)$. <i>Step I</i> minimizes $a_t(S)$. <i>Step II</i> computes the next bound $a_{t+1}(S)$	20
1.11	<i>K-means as linear bound optimization:</i> As obvious from (1.1), the bound $a_t(S)$ in Theorem 1 is a unary function of S . KM procedures (1.18) correspond to optimization of linear auxiliary functions $a_t(S)$ for KM objectives.	21
1.12	Weakly-supervised segmentation with different forms of weak supervision.	23
1.13	A synthetic example of semi-supervised learning with labeled and unlabeled data for binary classification. Given unlabeled data, the test point (?) is expected to be of positive class though it is closer to negative samples.	24

1.14	We study optimization in image segmentation with particular focus on combining these techniques in a principled way. For example, we jointly optimize MRF energy and clustering criterion in our Kernel Cut method [223]. Our regularized loss framework [224] bridges deep CNN segmentation and shallow segmentation methods including MRF and clustering.	26
2.1	Illustration of how pseudo-bound optimization framework updates current solution from S_t to S_{t+1} . Instead of optimizing only one auxiliary function $A_t(S)$, we explore a parametric family of pseudo-bounds. The best solution with the minimum original energy is chosen among the set of all global minima for the family. They correspond to breakpoints of parametric maxflow method [125] that can be efficiently explored in polynomial time. By optimizing the family of pseudo-bounds, larger decrease in energy is achieved ($S_t \rightarrow S_{t+1}$). [Best viewed in color]	31
2.2	Matching target foreground color distribution using auxiliary cuts [18], fast trust region [91] and pPBC. pPBC achieves the lowest energy.	33
2.3	Pseudo-bound families for two cardinality functions. Auxiliary functions are red.	36
2.4	pPBC-T : Pseudo-bounds (purple) and auxiliary functions (red) of non-submodular potential $m_{pq}s_p s_q$ for current configuration $s_{p,t} = 0, s_{q,t} = 0$ (left) and $s_{p,t} = 0, s_{q,t} = 1$ (right).	39
2.5	Left: interactive segmentations with BCD (GrabCut) or pPBC from different initialization (ellipses). Proposed pPBC method is more robust to inferior initialization. Right: unsupervised figure-ground segmentation with pPBC. Average color is shown.	41
2.6	Scatter plots; error rates versus energies for 500 solutions of BCD and pPBC.	41
2.7	Error rates and speed on GrabCut dataset for GrabCut [200], Dual Decomposition (DD) [234], One-Cut [220] and our pPBC method.	42
2.8	Segmentation with the curvature regularization model in [72].	44
2.9	Deconvolution results. Top row: noise $\sigma = 0.10$, bottom row: $\sigma = 0.2$	44

3.1	Segmentation of 5D image data (a). For higher-dimensional features, regularized model-fitting [263, 60] becomes sensitive to local minima, <i>e.g.</i> <i>grabcut</i> [200] fitting RGBUV histograms (b). Spectral clustering like <i>normalized cut</i> (NC) [211] is scalable to high dimensional features, but it is known for splitting regions (c) or lack of regularity (d). Our <i>kernel cut</i> (e) combines <i>kernel clustering</i> over arbitrary features with standard regularization in the image domain, see energy (3.1).	47
3.2	Our <i>Kernel Cut</i> approach to minimizing energy (3.1). Standard (MRF) regularization solvers can easily integrate our linear <i>kernel</i> or <i>spectral</i> bounds for the clustering term (Sec.3.3,3.4) producing an iterative <i>bound optimization</i> for (3.1).	50
3.3	<i>Model fitting (pKM)</i> (3.13) <i>vs kernel K-means (kKM)</i> (3.21). Histogram fitting converges in one step assigning initially dominant bin label (a) to all points in the bin (b): energy (3.13,3.14) is minimal at any volume-balanced solution with one label inside each bin [114]. Basic and elliptic K-means (one mode GMM) under-fit the data (c,d). Six mode GMMs over-fit (e) as in (b). GMMs have local minima issues; ground-truth initialization (f) yields lower energy (3.13,3.14). Kernel K-means (3.20,3.21) with Gaussian kernel k in (h) outperforms <i>pKM</i> with distortion $\ \cdot\ _k$ in (g) related to K-modes or mean-shift (<i>weak kKM</i> , see Sec.3.2.2).	52
3.4	Without edge alignment (3.2) GMM-fitting [200] shows stronger data overfitting compared to kernel clustering [211].	60
3.5	Equivalence of kernel clustering methods: <i>kernel K-means (kKM)</i> , <i>average distortion (AD)</i> , <i>average association (AA)</i> based on Roth et al. [197], see (3.29), (3.30). Equivalence of these methods in the general <i>weighted</i> case is discussed in [221, Appendix A, Fig.33].	64
3.6	Example: concave function $\tilde{e}(X) = -\frac{X'X}{1'X}$ for $X \in [0,1]^2$. Note that convexity/concavity of similar rational functions with quadratic numerator and linear denominator is known in other optimization areas, <i>e.g.</i> [27, p.72] states convexity of $\frac{x^2}{y}$ for $y > 0$ and [12, exercise 3.14] states convexity of $\frac{(v'X)^2}{w'X}$ for $w'X > 0$	73
3.7	Typical energy evolution wrt different moves and frequency of bound updates. α - <i>expansion</i> updates the bound after a round of expansions, α - <i>expansion*</i> updates the bound after each expansion move. Initialization is a regular 5×5 grid of patches.	77

3.8	Interpreting our linear bounds for E_A term in (3.1) via K -means: optimization of the <i>spectral</i> bound (alone) is equivalent to K -means algorithm over approximately isometric data embeddings in \mathcal{R}^m for $m \leq \Omega $, see Sec. 3.4. As m approaches $ \Omega $, the isometry becomes more accurate and our approximate spectral bound for E_A reduces to the exact <i>kernel</i> bound. While relations between E_A and K -means were known for $m = K$ [211] (as a heuristic, see Sec.3.4.3) and $m = \Omega $ [197, 9, 64] (as energy equivalence), we establish it in a new bound optimization context essential for our work.	79
3.9	Eigen decompositions for kernel matrix \mathcal{K} (a) and its rank m approximation $\tilde{\mathcal{K}}$ (b) minimizing Frobenius errors (3.56) [56]. Decompositions (a,b) give explicit embeddings (3.55,3.58) isometric to the kernels, as in the Mercer theorem. One specific example for the Gaussian kernel is in Fig. 3.10.	80
3.10	Low-dimensional Euclidean embeddings (3.58) for $m = 2$ and $m = 3$ in (c,d) are approximately isometric to a given affinity matrix (b) over the data points in (a). The approximation error (3.57) decreases for larger m . While generated by standard MDS methodology [56], it is intuitive to call embeddings ϕ in (3.55) and (3.58) as (exact or approximate) <i>isometry eigenmap</i> or <i>eigen isomap</i>	81
3.11	Spectrum of eigenvalues of typical kernel matrices for synthetic data (top row) or real image color (bottom row). This helps us to select approximate embedding so as to have small approximation error (3.57). For example, with fixed width gaussian kernel in (a), it suffices to select a few top eigenvectors since the remaining eigenvalues are negligible. Note that the spectrum elevates with increasing diagonal shift δ in (3.60). In principle, we can find the optimal shift for a given number of dimensions m to minimize approximation error.	87
3.12	For data and affinity matrix in Fig. 3.10, we run weighted K-means with our approximate embedding. The approximation errors $\ \mathcal{K} - \tilde{\mathcal{K}}\ _F^2 / \ \mathcal{K}\ _F^2$ for 3, 6, 10 and 50 dim. embedding are 58%, 41%, 27% and 3% respectively. We compute weighted K-means energy (up to a const) and normalized cuts energy for solution obtained at each iteration. We observed that normalized cuts energy indeed tends to decrease during iterations of K-means. Even 10 dim. embedding gives good alignment between K-means energy and normalized cuts energy. Higher dimensional embedding gives better energy approximation, but not necessarily better solution with lower energy.	88

3.13	Sample results on BSDS500. Top row: spectral clustering. Middle & Bottom rows: our Kernel & Spectral Cuts.	93
3.14	Segmentation using our kernel cut with label cost. We experiment with increasing value of label cost h_k for each label (from left to right)	94
3.15	Incorporating group prior achieves better NMI for image clustering. Here we use tags-based group prior. Our method achieved better NMI when more images are tagged. The right plot shows how the weight of bin consistency term affects our method.	95
3.16	Illustration of robustness to smoothness weight.	96
3.17	Average error vs. regularization weights for different variants of our Kernel-Cut on the GrabCut dataset.	97
3.18	Our method aKKM is robust to choice of K while GrabCut is sensitive to bin size for histograms.	98
3.19	Sample results for GrabCut and our kernel cut with fixed width Gaussian or adaptive width KNN kernel, see Tab. 3.8.	99
3.20	Sample results for BJ [30], GrabCut [200], and our kernel cut for adaptive KNN kernel, see Tab. 3.9.	100
3.21	Visualization of a pixel's K-Nearest-Neighbours for RGB feature (left) or RGBXY feature (right).	100
3.22	Error on Multi-objects dataset. We vary spatial bin-size for GrabCut and weight β in $[l, a, b, \beta X, \beta Y]$ for Kernel Cut. The connection range is the average geometric distance between a pixel and its k^{th} nearest neighbor. The right-most point of the curves corresponds to the absence of XY features. GrabCut does not benefit from XY features. Kernel Cut achieves the best error rate of 2.9% with connection range of 50 pixels. . . .	101
3.23	Multi-label segmentation for similar objects.	101
3.24	The average errors of GrabCut and Kernel Cut methods over 64 images selected from NYUv2 database [169].	102
3.25	RGBD+XY examples. The first two rows show original images wit bounding box and color-coded depth channel. The third row shows the results of Grabcut, the forth row shows the results of Kernel Cut. The parameters of the methods were independently selected to minimize their average error rates over the database.	102

3.26	Motion segmentation using our framework for the sequence <i>horses01</i> in FBMS-59 dataset [34]. Motion feature alone ($\mathbf{M}+\mathbf{XY}$ in (c)) is not sufficient to obtain fine segmentation. Our framework successfully utilize motion feature (optical flow) to separate the horse from the barn, which have similar appearances. See supplementary material for results on the video. . .	103
3.27	Multi-label motion segmentation using our framework for the sequence <i>ducks01</i> in FBMS-59 dataset [34]. This video is challenging since the ducks have similar appearances and even spatially overlap with each other. However, different ducks come with different motions, which helps our framework to better separate individual ducks. See supplementary materials.	104
3.28	Motion segmentation for image 000079_10 from <i>KITTI</i> [161] dataset. The first row shows the motion flow. Black color codes the pixels that do not have motion information. The second row shows color-based segmentation. The third row shows motion based segmentation with location features. We also tried $\mathbf{M}+\mathbf{XY}$ segmentation, but it does not work as well as $\mathbf{MXY}+\mathbf{XY}$ above. The results for $\mathbf{RGBMXY}+\mathbf{XY}$ were not significantly different from $\mathbf{MXY}+\mathbf{XY}$	105
4.1	We approach the problem of weakly-supervised segmentation (a) with scribbles through principled techniques in semi-supervised learning (b).	108
4.2	Different forms of weak annotations for training semantic segmentation. . . .	113
4.3	Proposals from interactive segmentation algorithms with seeds.	114
4.4	<i>Synthetic segmentation example</i> for grid and dense CRF (Potts) models: (a) intensities $I(x)$ on 1D image. The cost of segments $S^t = \{x \mid x < t\}$ with different discontinuity points t according to (b) nearest-neighbor (grid) Potts and (c) larger-neighborhood (dense) Potts. The latter gives smoother cost function, but its flatter minimum may complicate discontinuity localization.	118
4.5	<i>Real "shallow" segmentation example</i> for sparse (b) and dense (c) CRF (Potts) models for image with seeds (a). Sparse Potts gives smoother segment boundary with better edge alignment, while dense CRF inference often gives noisy boundary.	119
4.6	Visualization of the gradient for different losses. The negative (positive) gradients are coded in red (yellow). For example, negative gradients on the dog drives the network to predict "dog" for these pixels. Also, the dog pops out in the gradient map.	125

4.7	Examples on PASCAL VOC <i>val</i> set. Kernel cut as regularization loss gives qualitatively better result than that with normalized cut loss. We found kernel cut results to have better edge alignment.	126
4.8	Similar to [147], we shorten the scribbles. With length zero (clicks) is the most challenging case. Right plot shows mIOUs when train with shorter scribbles.	127
4.9	Examples for supervision with image-level labels (tags). We train using the seeding loss, expansion loss in SEC [121] and our CRF loss. Similar segmentation is obtained yet we avoid any iterative mean-field inference for dense CRF.	129
4.10	Training progress of ADM and gradient descend (GD) on Deeplab-MSc-largeFOV. Our ADM for the grid CRF loss with α -expansion significantly improves convergence and achieves lower training loss. For example, first 1,000 iterations of ADM give grid CRF loss lower than GD's best result. . .	130
4.11	Example segmentations (Deeplab-MSc-largeFOV) by variants of regularized loss approaches. Gradient descent (GD) for grid CRF gives segmentation of poor boundary alignment though grid CRF is part of the regularized loss. ADM for grid CRF significantly improves edge alignment and compares favorably to dense CRF based method.	133

Chapter 1

Introduction

1.1 Image Segmentation



(a)interactive segmentation; (b)motion segmentation; (c)semantic segmentation.

Figure 1.1: Segmentation problems of different kinds [200, 177, 74].

Segmentation is the task of decomposing an image to disjoint subsets of pixels. Each subset, a.k.a. *segment*, is usually assigned a discrete label. Fig. 1.1 shows various types of segmentation problems, including interactive segmentation given user input scribbles, motion segmentation of a moving object, and semantic segmentation with a predefined set of labels (e.g. car, horse and person). Other kinds of segmentation include, for example, superpixel segmentation [2] and instance segmentation [101].

Segmentation has many applications in autonomous driving, image editing, remote sensing, augmented reality, and medical image analysis etc. For example, interactive segmentation [200, 10] in image editing software enables faster content creation. Autonomous driving relies on road segmentation [179] to perceive the environment in order to navigate. Automatic segmentation of the building footprint in remote sensing [61] allows faster urban planning and disaster management. Another example is organ segmentation of medical images that helps to identify diseases.

Segmentation is an active and fundamental area of research in computer vision. It often serves as a building block for many other computer vision tasks. It has also been studied jointly with other problems such as optical flow [209], reconstruction [134], and tracking [162]. These tasks are complementary in nature and segmentation plays a critical part.

Unlike image classification, segmentation is a dense labeling problem, which should consider *geometry*. Is the boundary smooth? Does the shape of a segmentation look like a target object e.g. a horse? Are the segments connected rather than fragmented? These are natural properties and constraints in terms of the geometry of segmentation. Some of the properties have been modelled via graphical models, for instance, boundary smoothness as pairwise Markov Random Field [30].

Often in segmentation tasks, we only have weak supervision or annotations. For instance, in interactive segmentation, the user provides scribbles or bounding boxes for the objects of interest. For video segmentation, usually only the first frame is labeled, and the goal is to propagate the labeling to all subsequent frames. Weakly supervised semantic segmentation given scribbles, boxes, or image-level tags attracts a lot of interest since it eliminates the necessity of full labeling, which is laborious to obtain. A good algorithm for segmentation should make the best use of available annotations and reduce the amount of supervision needed. It also helps to incorporate prior knowledge into segmentation such as size, shape, and color distribution about the target segments.

While deep Convolutional Neural Networks achieve record-breaking accuracies, segmentation is far from being a solved problem. Firstly, the current state-of-the-art CNN relies on full supervision with tens of thousands of training images. Such approach of training is not scalable. Secondly, CNN segmentation generalizes poorly across different domains for the same set of classes. Thirdly, geometry and structure of segmentation is not explicitly modeled and incorporated in CNN segmentation. Lastly, video segmentation is largely under-explored since CNN is computationally expensive even for a 2D image.

This work focuses on unsupervised and weakly-supervised segmentation techniques. We are particularly interested in the formulation and optimization of image segmentation based on *regularization*.

notation	definition	range
S_p	segment (label) assignment for given pixel $p \in \Omega$	$\{1, \dots, K\}$
S_c	$(S_p p \in c)$ - labeling of pixels in <i>factor</i> $c \subseteq \Omega$	$\{1, \dots, K\}^{ c }$
S	$(S_p p \in \Omega)$ - segmentation or labeling of all points	$\{1, \dots, K\}^{ \Omega }$
S_p^k	indicator for “ p is in segment k ”, <i>i.e.</i> $S_p^k \equiv [S_p = k]$	$\{0, 1\}$
S^k	k -th segment , that is, subset $\{p \in \Omega S_p = k\}$ or the indicator vector $(S_p^k p \in \Omega)$	power set $\mathcal{P}(\Omega)$ or $\{0, 1\}^{ \Omega }$
$S^{k'}$	transpose of vector S^k , <i>i.e.</i> $S^{k'} \equiv (S^k)^T$	$\{0, 1\}^{ \Omega }$
S_t	segmentation at iteration t	same as S
$S_\theta(I)$	Softmax output of a CNN parameterized by θ	$[0, 1]^{ \Omega \times K}$

Table 1.1: Frequently used notations for segmentation in this thesis.

1.2 Notation and Conventions

Here we summarize notations commonly used in this thesis. We use notation applicable to either image segmentation or general data clustering. Let Ω be a set of pixels, voxels, or any other points p . For example, for a 2D image Ω could be a subset of regularly spaced points in \mathcal{R}^2 . Set Ω could also represent data points indices. We assume that every $p \in \Omega$ comes with an observed feature I_p . For example, I_p could be a greyscale intensity in \mathcal{R}^1 or an RGB color in \mathcal{R}^3 . The number of segments/clusters/labels is denoted by K . For instance, $K = 2$ for interactive foreground/background segmentation. By convention, the background is labeled 0.

Our notation describing segmentation of Ω is summarized in Table 1.1. We use the following standard notation: $\{\cdot\}$ stands for sets or subsets, (\cdot) stands for ordered collections or vectors, and $[\cdot]$ is used in the context of intervals, matrices, or *Iverson brackets*¹.

We let $A = [A_{pq}] \in [0, 1]^{|\Omega| \times |\Omega|}$ be an affinity matrix between data points/pixels. The degree of point p is $d_p = \sum_q A_{pq}$ and $D \in \mathcal{R}^{|\Omega| \times |\Omega|}$ is a diagonal matrix with $D_{pp} = d_p$.

¹Iverson brackets $[\cdot]$ enclosing a logical proposition, e.g. $[S_p = k]$, return 1 or 0 depending on true or false value of this proposition.

1.3 Overview of Segmentation Techniques

Over the last few decades, lots of heuristics, algorithms, and mathematical models have emerged for image segmentation [37, 211, 42, 30, 53, 200, 185, 6, 130, 150, 223, 101]. Fig. 1.2 shows a word cloud of common techniques and models for segmentation. Notable ones are Chan-Vese [42], Graph Cut [30], Spectral Method [211], Dense CRF [130], FCN [150], and Mask R-cnn [101] etc.

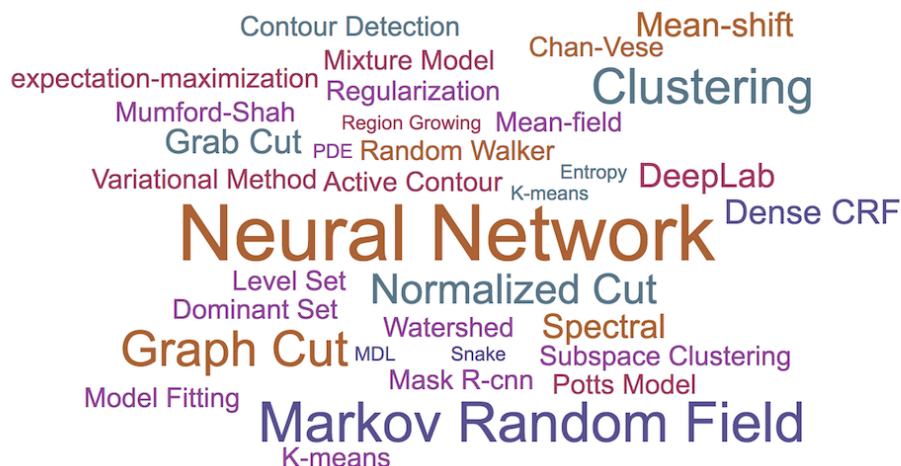


Figure 1.2: Segmentation Techniques.

There is no unified taxonomy due to the diversity of these techniques. Here we discuss segmentation techniques from different perspectives.

- **optimization v.s. non-optimization methods:** Often, segmentation is driven by optimization. For example, Graph Cut [30] minimizes a joint objective of unary data fidelity term and smoothness term. The spectral method minimizes pairwise clustering criterion e.g. normalized cut [211]. However, some algorithms are motivated by a dynamic or mechanism, and it's not clear what the objective is. For instance, a dominant set for segmentation [185] is sought by replicator dynamics arising in evolutionary game theory. Mean-shift [53] finds the modes in the probability distribution of image features, and we obtain the final segmentation by merging heuristic. K-means is both optimization and non-optimization method since the dynamic of the K-means algorithm (updating cluster center & cluster assignment) indeed minimizes the sum of squared distances. Here we focus on optimization methods.

- **shallow v.s. deep methods:** In the era of deep learning, neural networks, in particular fully convolutional neural networks, have revolutionized image segmentation. CNNs give record-breaking results for interactive segmentation [246], semantic segmentation [150] and instance segmentation [101]. Shallow² methods are largely overlooked nowadays. In this thesis, we combine shallow and deep methods, including Markov Random Field, Kernel Clustering and CNN, see Sec. 1.4.3. We show that shallow and deep methods are mutually beneficial.
- **unsupervised v.s. supervised methods:** CNN segmentation is typically fully supervised and needs thousands of labeled images. The more the supervision, the better is the segmentation. In contrast, methods like normalized cut [211] are unsupervised and rely purely on low-level features and cues. Recently, weakly-supervised CNN segmentation has attracted lots of interest with various forms of supervision, including scribbles [147, 219, 224], bounding boxes [116], polygons [38], and image-level tags [121]. In the lack of supervision, regularization is useful to yield better segmentation, as discussed below. Also, we bring principled techniques from semi-supervised learning [265, 44] to weakly-supervised segmentation, see discussion in Sec. 1.4.2.
- **regularized v.s. unregularized methods:** Regularization in the context of segmentation is very different from that in machine learning which prevents from overfitting. A standard regularization for segmentation is the Potts model [190] that can be minimized by MRF method [31, 130], level-set method [42], and convex relaxation approach [189] etc. Shape regularization and priors [119, 92] were modelled in an energy minimization based method. Chew *et al.* [48] investigated semi-supervised normalized cut with must-link and cannot-link constraints. CNN Segmentation benefits from regularization as shown in previous work that incorporated regularization as post-processing [45], trainable layers [207, 260], or losses [224]. Regularization helps fully-supervised and weakly-supervised CNN segmentation.

In the following, we briefly review segmentation techniques that are most relevant to the work here. We first start with the simple K-means algorithm in Sec. 1.3.1. Then we review (nonlinear) Kernel Clustering, Markov Random Filed, and Convolutional Neural Network in Sec. 1.3.2, 1.3.3, and 1.3.4.

²In this thesis, by "shallow" we mean any method unrelated to deep learning. An example of shallow methods is the seminal work of (iterated) graph cut [30, 200].

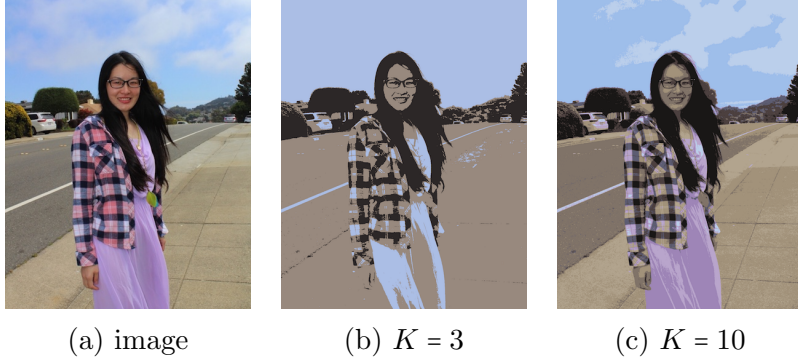


Figure 1.3: K-means segmentation for different number of segments.

1.3.1 A Toy Example: K-means

K-means [153] is one of the simplest segmentation algorithms. K-means clustering, *a.k.a.* Lloyd’s algorithm [149], first initializes cluster centers and then iterates between the following two steps until converge.

- For each data point, compute its distances to the K cluster centers. Assign the point to the cluster corresponding to the closest distance;
- Update the cluster centers to be the arithmetic mean of data points of each cluster.

This iterative algorithm minimizes the sum-of-square distances between data points and cluster centers. Let Ω be the set of pixels in image I , $I_p \in R^3$ be the color for pixel p , then the K-means objective on color is,

$$F_{km}(S, \mathbf{m}) := \sum_{p \in \Omega} \|I_p - m_{S_p}\|^2 \quad (1.1)$$

where $S = \{S_p\} \in \{1, \dots, K\}^{|\Omega|}$ is the discrete labeling and $\mathbf{m} = \{m_k\}$ are the cluster centers or means. Norm $\|\cdot\|$ denotes the Euclidean metric.

K-means segmentation basically gives color quantization, see Fig. 1.3. The popular SLIC superpixel [2] is based on K-means on 5-d RGBXY input features. The Chan-Vese model [42] minimizes a binary version of the sum of squared distance (1) with $K = 2$.

1.3.2 Kernel Clustering (KC)

Clustering and segmentation are largely synonyms. Clustering algorithms are categorized as linear or nonlinear. K-means is limited to clusters that are linearly separable while many other algorithms, including mean-shift [53], linkage based method [96], kernel/pairwise clustering [211], and dominant set [185], can give complex-shaped clusters. K-means is an example of *partitional clustering* that assigns each data point to one cluster. *Probabilistic clustering* such as Gaussian Mixture Model (GMM) determines the probability of each data point belonging to the clusters. The input to clustering algorithms are typically represented as vectors or graphs.

For data clustering, it is natural and common to assume that,

- data points in the same cluster should be *similar*;
- data points in different clusters should be *dissimilar*.

Different algorithms differ in terms of the distance metric and the criteria for what is good clustering. Here, we are particularly interested in (nonlinear) kernel/pairwise clustering. A seminal work is Normalized Cut (NC) [211] for image segmentation proposed by Shi & Malik.

$$F_{nc}(S) := \sum_k \frac{assoc(S^k, \Omega/S^k)}{assoc(S^k, \Omega)} \equiv K - \sum_k \frac{assoc(S^k, S^k)}{assoc(\Omega, S^k)}, \quad (1.2)$$

where $S^k = \{p | s_p = k\}$ is a cluster, $A = [A_{pq}] \in [0, 1]^{| \Omega | \times | \Omega |}$ is an affinity matrix, $assoc(S^k, \Omega \setminus S^k) = \sum_{p \in S^k, q \in \Omega/S^k} A_{pq}$ is the sum of associations between points in the cluster S^k and in its complement $\Omega \setminus S^k$. Normalized Cut (1.2) minimizes the "cut" between clusters. The normalization by $assoc(\Omega, S^k)$ encourages balanced clusters and helps to avoid a cut that isolates outlier. NC also maximizes the balanced association within each cluster, since it is equivalent to normalized association up to an additive constant K , see (3.38). Normalized cut for image segmentation first builds an affinity matrix A using Gaussian kernel on low level features (color, edge and texture) and then minimizes (1.2).

There are other criteria of kernel clustering such as averaged cut, average association, or ratio cut. They are all different combinations of ratio terms aiming to maximize the intra-cluster association or minimize the inter-cluster cut. In Chapter 3, we detail these criteria and discuss their optimization and biases that are not known before.



Figure 1.4: Spectral method for normalized cut segmentation.

Spectral methods for kernel clustering typically involve eigen-decomposition of the Laplacian matrix or the normalized Laplacian matrix, followed by discretization heuristics on the (weighted) eigenvectors, e.g. K-means. A version of the spectral method for Normalized Cut is outlined in Alg. 1. Fig. 1.4 visualizes the top eigenvectors.

Algorithm 1: Spectral Method for Normalized Cut Clustering.

Input: Affinity matrix $A = [A_{pq}] \in [0, 1]^{|\Omega| \times |\Omega|}$; Number of clusters K ;

Output: Clusters $S^k = \{p | s_p = k\}$ for $k = 1, \dots, K$;

- 1: Compute the degree $d_p = \sum_q A_{pq}$, the degree matrix $D \in R^{|\Omega| \times |\Omega|}$ with $D_{pp} = d_p$, and the Laplacian matrix $L = D - A$;
 - 2: Find the top K eigenvectors \mathbf{u}^k of the eigen system $L\mathbf{u} = \lambda D\mathbf{u}$;
 - 3: Construct a matrix $\tilde{\Phi}$ of size $\Omega \times K$ where each column is the eigenvector \mathbf{u}^k .
 - 4: Run K-means on rows of $\tilde{\Phi}$ which are of K dimension.
 - 5: **return** S_k ;
-

It is shown in [211] that In the special case of binary clustering with $K=2$, the eigenvectors are the global optimum of normalized cut in the relaxed space $[0, 1]^{|\Omega| \times K}$. Hence a discretization heuristic like K-means is necessary. It is supposed that the low dimensional and nonlinear embeddings based on eigenvectors make the data linear separable. Various spectral clustering methods differ in terms of how to construct the affinity matrix, what criteria to minimize, which eigensystem to solve, and the discretization heuristics. For example, the affinity matrix can be based on a Gaussian kernel with fixed bandwidth or from K nearest neighbors [235], which is robust to density variance of the data. Some variants find the eigenvectors of the unnormalized Laplacian L .

Normalized cut is widely used for image segmentation [211], image clustering, and general graph clustering. Normalized cut and other graph clustering criteria are shown to be equivalent to kernel k-means [9, 64, 197]. This connection gives an alternative optimization method beside spectral method. Spectral clustering and Kernel K-means are also closely related to graph embedding and dimensionality reduction techniques such as kernel PCA [205], Iso Map [225], and Laplacian eigenmap [15]. In this thesis, we dive into the formulation, optimization, and extension of spectral clustering, kernel K-means, and dimensionality reduction. We revisit their connections which allows us to gain insights, reveal previously unknown biases, and develop new formulation and optimization method for kernel clustering.

The number of clusters K needs to be specified as a *prior* for normalized cut. It is also unclear how to *validate* the clustering obtained since typically there is no ground-truth for such an unsupervised task. Here, we focus on optimization algorithms for kernel clustering rather than model selection, validation [35], or other theoretical aspects [19] of clustering. Alg. 1 also needs a predefined affinity matrix A , which is often based on simple Gaussian [211, 235] or KNN kernel [256]. It is possible to learn the affinity matrix [154], but in this work we assume the affinity matrix to be given.

Kernel clustering is not a solved problem though there exists theoretically sound algorithm e.g. Alg. 1. It is often necessary to incorporate constraints and regularization to kernel clustering.

- grouping constraints: A subset of points should be in the same cluster;
- *must-link* constraints: two points should be in the same cluster;
- *cannot-link* constraints: two points should be in different clusters;
- boundary smoothness regularization: spectral clustering for image segmentation tends to partition uniform regions, see an example result in Fig. 1.4 (b).

Different formulations and methods have been proposed for constrained or regularized kernel clustering [252, 245, 132, 73, 51, 48]. For example, a different eigen problem is solved for semi-supervised normalized cut with must-link and cannot-link constraints [48]. In this work, we give a principled formulation of kernel clustering and its optimization allowing a wide range of constraints and regularizations.

1.3.3 Markov Random Field (MRF)

Markov Random Field, *a.k.a* undirected graphical model, is a type of probabilistic graphical model [122] which in general encodes probabilistic dependencies between variables represented by graph nodes.

Definition 1 (Markov Random Field). *A Markov Random Field is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E} \subset \mathcal{V} \times \mathcal{V})$ where nodes $\mathcal{V} = \{S_1, \dots, S_{|\Omega|}\}$ represent random variables, edges \mathcal{E} represent dependencies between variables, and a node is independent of all other nodes except its neighbors, which is also known as Markovian Property.*

MRF is a popular model in computer vision [23]. Markovian property is natural for 2D image grid due to a sense of spatial locality and regularity. Many computer vision problems like segmentation and optical flow are ill-posed. *Regularization* is introduced to eliminate the ambiguity of a possibly infinite number of solutions and incorporate *structure* and *prior* for the output space. Markov Random Filed, as a framework for regularization, has been studied extensively in computer vision and image processing [23, 113]. Note that unlike image classification, segmentation is a dense prediction task that outputs some quantities or labels for all pixels. For such dense prediction problems, MRF is a powerful tool to incorporating regularization, imposing constraints, and making the output structured.

The joint probability distribution of variables $S_1, \dots, S_{|\Omega|}$ in MRF is a *Gibbs distribution* that can be factorized over cliques \mathcal{C} ,

$$\mathcal{P}(S) \propto \prod_{c \in \mathcal{C}} \mathcal{P}_c(S_c), \quad (1.3)$$

where each clique $c \in \mathcal{C}$ is a predefined subset of pixels and S_c is the set of variables S_p associated with the clique c . The cliques \mathcal{C} can be pairwise or higher order, see illustration in Fig. 1.5. The *maximum a posterior* (MAP) estimation of MRF is the solution that maximize the likelihood (1.3),

$$S^* = \arg \max_S \mathcal{P}(S). \quad (1.4)$$

The maximum of multiplicative probability $\mathcal{P}(S)$ can be alternatively sought by minimization of the additive energies³, which is the negative log of probability, i.e. $E_c(S_c) = -\log \mathcal{P}(S_c)$.

³Due to the connection to statistical physics, such objective is referred as "energy" in the literature.

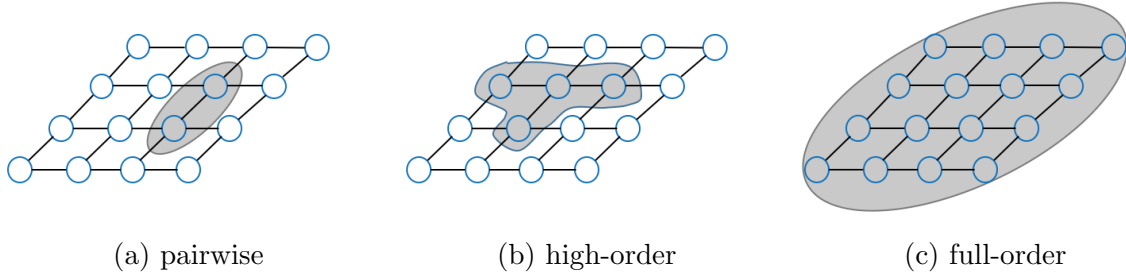


Figure 1.5: Factors/Cliques of different orders for 2D image.

Definition 2 (Energy Minimization). *The problem of finding the labeling S^* of minimum cost for objective $F_{mrf}(S)$ is called energy minimization, which gives the MAP solution (1.4) to a Markov Random Field.*

$$S^* = \arg \min_S F_{mrf}(S), \text{ for } F_{mrf}(S) := \sum_{c \in \mathcal{C}} E_c(S_c). \quad (1.5)$$

It is easy to see that minimizing the energy (1.5) is equivalent to finding the MAP estimation of a factorizable Gibbs distribution, see [122] for such a probabilistic motivation. However, energy (1.5) can also be motivated intuitively, e.g. towards smoothness in segmentation. In the last few decades, different formulations, optimization algorithms, and learning strategies have been proposed for MRF. Here we are particularly interested in the optimization of MRF/CRF⁴ energies in the context of image segmentation.

A widely studied energy for computer vision is the following that consists of unary potentials $\phi_p(S_p)$ and pairwise potentials $\phi_{pq}(S_p, S_q)$.

$$F_{pmrf}(S) := \sum_{p \in \Omega} \phi_p(S_p) + \sum_{p, q \in \mathcal{N}} \phi_{pq}(S_p, S_q). \quad (1.6)$$

Modeling of MRF

The energy (1.5) can be defined on pairwise, high-order, or even full-order cliques/factors, as illustrated in Fig. 1.5. For instance, standard graph cut based segmentation [30, 200] combines unary log likelihoods and pairwise smoothness term.

$$F_{gc}(S; \theta^0, \dots, \theta^{K-1}) := \sum_{p \in \Omega} -\log P(I_p | \theta^{S_p}) + \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q], \quad (1.7)$$

⁴A Conditional Random Field (CRF) models the conditional dependencies between latent variables and observed data. For simplicity, we use the term MRF and CRF interchangeably in this paper.

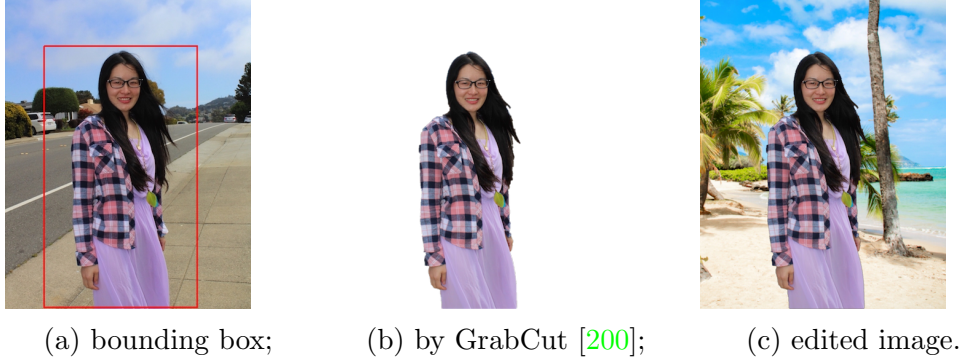


Figure 1.6: GrabCut (Iterated Graph Cut) [200, 30] for interactive segmentation with box.

where $\theta^0, \dots, \theta^{K-1}$ are appearance models for the K segments, a set of pairwise factors \mathcal{N} includes *edges* $c = \{pq\}$ between pairs of neighboring nodes, $[\cdot]$ are *Iverson brackets*, and w_{pq} is a discontinuity penalty between p and q . Penalty w_{pq} can be a constant or may be set by a decreasing function of intensity difference $\|I_p - I_q\|$ attracting the segmentation boundary to image contrast edges [30]. The term $w_{pq} \cdot [S_p \neq S_q]$ measures contrast-sensitive boundary length and prefers edge alignment. The neighborhood system \mathcal{N} is typically a 4 or 8 connected grid. This is similar to the image-based boundary length in geodesic contours [37, 28]. Fig. 1.6 shows an example of interactive segmentation and editing, the energy of which can be efficiently optimized by graph cut [30].

High-order MRFs involve high-order factor/cliue of more than two pixels. In general, higher-order potentials are more expressive than pairwise potentials in terms of incorporating regularization and prior. However, they are often more challenging to optimize. The basic Potts model in (3.2) prefers two nearby pixels that are similar to take the same label. P^n Potts model [120] generalizes the basic Potts model and encourages consistent labeling among a set of pixels. For example, the consistency potential is defined on each superpixel [120] or bins of pixels [220, 183] with the same color/features. Also, the minimization of log likelihoods in standard MRF based segmentation can be interpreted as entropy minimization [220], which is essentially high-order. High-order MRFs have also been utilized in other computer vision tasks besides segmentation. Woodford *et al.* [242] proposed second order smoothness prior for stereo reconstruction which relies on triple cliques. High-order models like *filed-of-experts* [196] have been demonstrated useful for denoising and inpainting.

Shape priors and other geometric constraints often require high-order potentials. For instance, objects may have convex shape in some cases. The convexity shape prior [92]

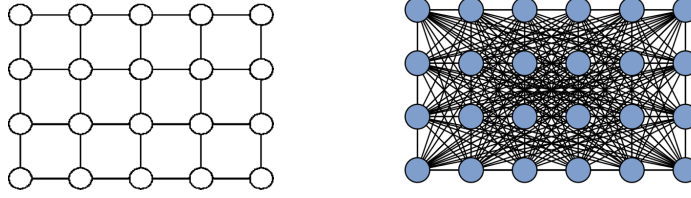


Figure 1.7: Typical CRF models used for image segmentation. Left: Sparse or grid CRF [30]; Right: Dense CRF [130].

has been formulated as MRF potentials on triple clique that can be optimized by discrete algorithms. An issue with the Potts model (3.2) is its shrinking bias towards shorter boundaries. Curvature regularization avoids such bias and works better than length-based regularization (3.2) for thin structures. It is shown that squared curvature can be approximated by high-order potentials [173]. Some seemingly complex prior such as star-shape prior [229] can be reduced to simple pairwise MRF for pixels along centered rays.

Worth mentioning are high-order potentials defined on all pixels, *a.k.a.* full-order⁵. A simple example is the volume prior [91] $\|\sum_{p \in \Omega} S_p - V_0\|^2$ that penalizes the deviation from a target volume V_0 . Volume is a moment term of order zero. Higher-order moments like 1st and 2nd moments control the center and aspect ratio of a segment. Gorelick *et al.* [91] formulated moment constraints as high-order MRF. Besides the target volume and moment, we may know about the appearance model, e.g. color distribution of the target object. A distribution matching term between a target distribution and the achieved distribution is high-order [91, 217] and can be optimized approximately. Another prior is connectivity. An object should be connected rather than fragmented. Whether a segment is connected or not is obviously full-order since we need to consider the connectivity for all pixels. Connectivity prior [233] has been incorporated to graph cut based optimization. Another example of full-order MRF defined for a single high-order factor $c = \Omega$ is label cost [60] that is a sparsity potential. In its simplest form, it penalizes the number of distinct segments (labels). Co-occurrence statistics [135] is introduced to measure how likely objects appear together in an image and is shown useful for semantic segmentation.

A popular MRF model for segmentation is the Dense CRF [130] with Gaussian potentials $\sum_{p,q \in \Omega} w_{pq} [S_p \neq S_q] = \sum_{p,q \in \Omega} e^{-\frac{\|I_p - I_q\|^2}{2\sigma^2}} [S_p \neq S_q]$, see Fig. 1.7. The regularization effects of grid vs. dense CRF are different [231].

⁵In this case, the local Markovian property doesn't hold. We still refer to as MRF by convention.

Inference of MRF

Here, we briefly review optimization/inference for discrete MRF, i.e. $S \in \mathcal{S} = \{0, \dots, K - 1\}^{|\Omega|}$. There are two kinds of inference problems, namely MAP inference and probabilistic inference [122]. MAP inference finds the most likely solution $S^* = \arg \max \mathcal{P}(S)$ which boils down to optimizing the energy (1.5) $S^* = \arg \min_S F_{mrf}(S)$. Probabilistic inference computes the marginal $\mathcal{P}_p(S_p) = \sum_{S \in \mathcal{S}, S'_p = S_p} \mathcal{P}(S)$ and finds the solution of max-marginals $S_p^* = \arg_{S_p} \max \mathcal{P}_p(S_p)$. This work is focused on MAP inference.

Pairwise MRF (1.6) is in general NP-hard to optimize, while limited cases can be optimized globally in polynomial time. If the neighboring system \mathcal{N} forms a chain or more generally a tree, then *dynamic programming* solves $\min_S F_{pmrf}(S)$ globally in the order of $K^2 \times |\Omega|$. For a graphical model with low treewidth, the junction tree algorithm [59] can be used to obtain the global optimum.

Dynamic programming on a chain is a special case of *belief propagation* [186, 247] (BP) for tree-structured MRF. Belief propagation works as beliefs are propagated from a node to its neighbors in a certain order. The min-sum algorithm and sum-product algorithm solve MAP inference and marginal inference exactly for a tree, involving two passes from leaves to root and from root to leaves (backtracking). BP also gives all the min-marginals in the forward pass from leaves to root and can be seen as re-parameterization of the energy parameters ϕ_p and ϕ_{pq} [124].

Often it is natural in computer vision to deal with graphs, e.g. 4-connected image grid. Loopy belief propagation (LBP) [80, 239, 76] passes messages on loopy graphs. The order in which to update the messages can be *parallel* or *sequential*. For example, in the parallel scheme, messages for each node are updated simultaneously. However, LBP is not guaranteed to converge, and its theoretical analysis is limited.

Besides belief propagation, another widely-used technique for MRF inference is graph cut⁶ [98, 31, 30]. For binary MRF of $K = 2$, if it holds that

$$\phi_{pq}(0, 0) + \phi_{pq}(1, 1) \leq \phi_{pq}(0, 1) + \phi_{pq}(1, 0) \quad \forall p, q, \quad (1.8)$$

then (1.6) is graph representable and can be optimized globally using *graph cut* [98, 31, 127]. This method constructs a graph with each node representing each pixel, together with two special nodes of *source* and *sink*. The cut of the constructed graph is equivalent to pairwise energy (1.6) with the corresponding labeling. Standard maxflow algorithms such as Ford-Fulkerson algorithm and push-relabel [86] find the mincut and optimal labeling S .

⁶This technique of mincut/maxflow is referred as "graph cut" in computer vision by convention. Here, we do not mean the general task of cutting or partitioning a graph.

The energy of a binary MRF, i.e. $K = 2$, can be thought as a set function $F(\mathbf{A})$ on the set $\mathbf{A} = \{p | S_p = 1\}$. Graph cut is an example of *submodular* set functions whose minimum can be found in strongly polynomial time⁷ [99, 206].

Definition 3 (Submodularity). *A set function $F : 2^{|\Omega|} \rightarrow \mathcal{R}$ is submodular if for every $\mathbf{A} \subset \Omega$, $\mathbf{B} \subset \Omega$,*

$$F(\mathbf{A} \cup \mathbf{B}) + F(\mathbf{A} \cap \mathbf{B}) \leq F(\mathbf{A}) + F(\mathbf{B}). \quad (1.9)$$

Equivalently, a set function F is submodular if for $\mathbf{A} \subset \mathbf{B} \subset \Omega$ and $p \in \Omega \setminus \mathbf{B}$,

$$F(\mathbf{A} \cup p) - F(\mathbf{A}) \geq F(\mathbf{B} \cup p) - F(\mathbf{B}). \quad (1.10)$$

It is proved that pairwise MRF with potentials satisfying (1.8) is submodular [128].

Optimization of general multi-label MRF is usually NP-hard. Special cases exist that allows global optimum in polynomial time. For example, if the pairwise potential $\phi(S_p, S_q)$ is in the form of $\phi(S_p, S_q) = g(|S_p - S_q|)$ with convex $g(\cdot)$, then its global optimum is found by maxflow/mincut on the graph proposed by Ishikawa [109].

Boykov *et al.* [31] developed move making algorithms, α -expansion and $\alpha\beta$ -swap, for MRF inference when the pairwise potentials $\phi(S_p, S_q)$ are *metric* and *semimetric*. α -expansion is an efficient algorithm with a constant approximation factor of the global optimum. $\alpha\beta$ -swap deals with more general energies that are semimetric. At each iteration of move making algorithms, a global optimum is sought by graph cut in a restricted search space in $\{0, \dots, K-1\}^{|\Omega|}$. For example, at each iteration of α -expansion, variables can choose to stay as their original labels or be updated to α . For $\alpha\beta$ -swap, those variables whose labeling is α or β can swap their labels.

Graph cut and move making [31] is the first widely-used algorithm for MRF inference in various applications such as stereo, optical flow, and segmentation. It gives much better results than earlier work including iterated conditional modes (ICM) [21, 68] and simulated annealing (SA) [82]. ICM [21, 68] is in a way analogous to move-making as finding optimum sequentially for each node. However, unlike α -expansion or $\alpha\beta$ -swap, the search space of ICM at each iteration is very limited, leading to a poor local minimum. SA [82] incorporated randomization by randomly updating a variable based on a probability distribution defined by the energies. However, SA is also restricted to optimizing one variable at a time and converges slowly in practice. Note that Boykov *et al.* [31] is not the first to use graph cut for computer vision problems. Greig [98] utilized maxflow/mincut for binary

⁷Though polynomial, the general algorithm for submodular minimization is usually impractical for computer vision problems with $|\Omega|$ in the order of millions.

image restoration and achieved global optimum. Roy and Cox [70] proposed a maxflow formulation of multi-label stereo. However, these work is restricted to binary energy or particular multi-label energy.

Move-making algorithms have been extended in many different ways. Veksler [230] proposed range move for truncated convex potentials which is common in computer vision. Another notable work along this line is the fusion move invented by Lempitsky *et al.* [142] that fuses multiple candidate solutions which comes from different algorithms maybe with different parameter settings. It is a meta-algorithm applicable to many problems.

An important class of methods for MRF optimization is based on linear programming (LP) relaxation and dual decomposition [12]. We first rewrite the discrete MRF energy in (1.6) as an integer linear program (ILP). We let $x_{p,i} = 1$ if $S_p = i$ and 0 otherwise. Similarly, $x_{pq,ij} = 1$ if $x_p = i, x_q = j$ and 0 otherwise. Then we have,

$$\begin{aligned}
\min_{\mathbf{X}} \quad & F(\Phi, \mathbf{X}) \equiv \sum_{p \in \Omega} \phi_{p,i} x_{p,i} + \sum_{pq \in \mathcal{N}} \phi_{pq,ij} x_{pq,ij} \equiv \langle \Phi, \mathbf{X} \rangle \\
\text{subject to} \quad & \sum_i x_{p,i} = 1, \forall p, \\
& \sum_j x_{pq,ij} = x_{p,i}, \forall p, \\
& x_{p,i} \in \{0, 1\}, \forall p \\
& x_{pq,ij} \in \{0, 1\}, \forall pq \in \mathcal{N},
\end{aligned} \tag{1.11}$$

where $\mathbf{X} = [\dots, x_{pi}, \dots, x_{pq,ij}, \dots]$ are binary variables $x_{pi}, x_{pq,ij}$ to be determined, and $\Phi = [\dots, \phi_{pi}, \dots, \phi_{pq,ij}, \dots]$ are MRF parameters $\phi_{pi}, \phi_{pq,ij}$.

ILP problem like (1.11) is NP-hard in general. If we relax the integer constraints $x_{p,i} \in \{0, 1\}, x_{pq,ij} \in \{0, 1\}$ to the range of $[0, 1]$, i.e., $x_{p,i} \in [0, 1], x_{pq,ij} \in [0, 1]$, then it becomes the linear programming (LP) relaxation⁸ of (1.6), which is widely studied in many approximation algorithms. However, LP relaxation still involves many variables and constraints in typical computer vision applications. It cannot be solved in practice using off-the-shelf LP solver like CPLEX [1].

Decomposition is a general principle for optimization. An optimization problem that is difficult can be decomposed into many smaller problems that are easier to solve, and the solution for the original original problem is from the solutions of the subproblems.

⁸LP relaxation is one of many possible convex relaxations for MAP-MRF, see a review and detailed analysis in [133].

Wainwright *et al.* [236] considered a convex combination of trees parameterized as $\sum_m \Phi^m = \Phi$ and maximized the dual of LP relaxation.

$$\begin{aligned} \max \quad & \sum_m F^*(\Phi^m) \\ \text{subject to} \quad & \sum_m \Phi^m = \Phi, \end{aligned} \tag{1.12}$$

where $F^*(\Phi^m) = \min_{\mathbf{X}} F(\Phi^m, \mathbf{X})$ is the minimum for the subproblem. Tree reweighted message passing (TRW) algorithms [236, 124] maximize the dual which is a lower bound of the original problem $F(\Phi, \mathbf{X})$. Like any other dual method, the gap between the dual objective and the original objective indicates how good is the solution. TRW algorithms iterate between reparameterization via belief propagation on trees and averaging of min-marginals to give new Φ^m . The iteration scheme matters for convergence of the algorithm. Kolmogorov [124] proposed TRW-S which updates messages sequentially which never decreases the lower bound, unlike what happens when updating the messages in parallel [236]. Through convergent, TRW-S is not guaranteed to find the global optimum of the dual problem. However, for particular class such as pairwise MRF with submodular potentials, it gives *strong tree agreement* and hence global optimum of the original MAP-MRF problem. In general, TRW-S may converge to a point of *weak tree agreement* where the dual objective no longer increases.

Komodakis *et al.* proposed a general dual decomposition (DD) method [129] that gives theoretical and practical benefits compared to other message passing algorithms. Similar to TRW-S, the original MRF on general graph is decomposed to a combination of solvable subproblems. A Lagrange dual is maximized through projected supergradient descent. MRF-DD [129] is more powerful than TRW-S since it achieves the global maximum of the dual objective while TRW-S leads to local maximum. MRF-DD is also more general allowing different forms of decomposition while TRW-S is limited to tree based decomposition. Such generality allows tighter relaxations of the original MRF by different decomposition scheme and the DD framework [129] is applicable to high-order MRFs.

So far, we've briefly reviewed MRF inference techniques based on graph cut, belief propagation, LP relaxation, and dual decomposition. Kappes *et al.* [113] systematically compared modern inference techniques for discrete energy minimization. Thorough experiments with different approaches, e.g. α -expansion, LBP, TRW-S, QPBO, DD, are conducted on various MRF models. The code is publicly available in OpenGM⁹. It is beyond the scope of this section to review in detail. A tutorial of MRF inference is given by Savchynskyy [203].

⁹<http://hciweb2.iwr.uni-heidelberg.de/opengm/>

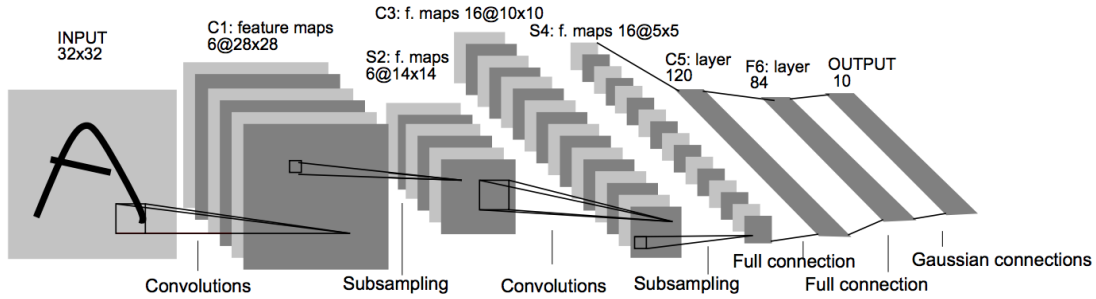


Figure 1.8: Architecture of LeNet-5 for digits recognition [138]. ©IEEE

1.3.4 Convolutional Neural Networks (CNN)

The above mentioned MRF and clustering method for segmentation are typically not related to deep learning [137]. We refer to them as "shallow" method for image segmentation. These shallow methods are driven by minimization of some objectives such as MRF energy or clustering criterion, while neural networks are trained to minimize the *empirical risk loss* w.r.t. lots of labeled data.

Deep learning [137, 106] has revolutionized computer vision [131, 102], natural language processing [52, 87], speech recognition [104] etc. Nowadays, deep learning dominates most of computer vision tasks, for example image classification [131, 102], tracking [237, 167], segmentation [45, 259, 101], optical flow [108, 214], human pose estimation [227] and image captioning [249].

Convolutional Neural Networks are widely used for computer vision. It is a type of multi-layer neural networks originally designed for visual recognition. LeNet [138] is the very first CNN successfully used for digit classification. Fig. 1.9 shows the architecture of LeNet which is composed of convolution layers, subsampling layers, and fully connected layers. The stacking of nonlinear layers yields very complex nonlinear transformation of an input image. A convolution layer has far less parameters than that of a fully connected layer. Convolution operation is linear w.r.t. the number of pixels. What's more, convolution is spatially invariant. The receptive field is enlarged by applying convolution several times.

CNNs for classification are trained to minimize a cross entropy loss. LeNet is relatively simple while recently developed network architectures such as GoogLeNet and ResNet are much deeper. However, a challenge with deeper network is optimization since the gradients are more likely to vanish or explode with deeper networks. The optimization of these

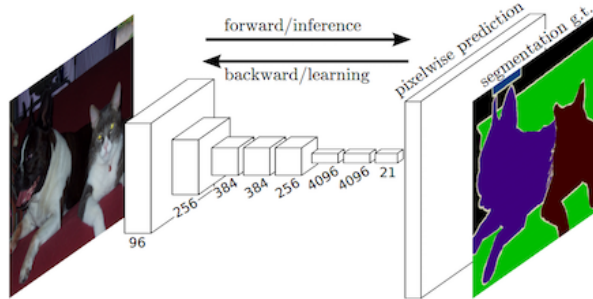


Figure 1.9: Fully convolutional neural networks for semantic segmentation [150]. ©IEEE

really deep network is feasible nowadays with advanced optimization and regularization techniques such as auxiliary losses [216], Dropout [213] and residual network [102].

In the era of deep learning, convolutional neural networks originally developed for classification also revolutionized the field of segmentation. The seminal work of Long *et al.* [150] is the first that trains a fully convolutional network for semantic segmentation, achieving accuracy significantly better than previous state-of-the-art. The observation is that a fully connected layer in a classification network, e.g. AlexNet [131], is also a 1×1 convolution. So the fully connected layer is removed and striding and pooling layers are used less frequently to yield a coarse feature map. The coarse feature maps are restored to the original resolution via e.g. deconvolution layer [150]. In order to utilize both high-level and low-level features, coarse and fine layers are concatenated and convolved to give combined features. An example of such network with *skip-connection* is FCN-8s [150]. Finally, pixel-wise segmentation in the form of probability distributions is obtained by a softmax layer on the scores.

CNNs for semantic segmentation are trained end-to-end by back-propagation. It requires a set of images $\{I_1, \dots, I_N\}$ and the corresponding ground truth labeling or mask $\{Y_1, \dots, Y_N\}$. The number of images N is typically in the order of thousands. Let $S_\theta(I) \in [0, 1]^{|\Omega| \times K}$ be the output of a segmentation network parameterized by θ and $S_p^{Y_p}$ be the probability that pixel p belongs to its ground-truth label Y_p . We minimize an *empirical risk loss* w.r.t. ground truth $Y \in \{1, \dots, K\}^{|\Omega|}$,

$$\min_{\theta} \frac{1}{N} \sum_{n=1}^N \ell(S_\theta(I_n), Y_n). \quad (1.13)$$

where $\ell(S_\theta, Y) = \sum_{p \in \Omega} -\log(S_p^{Y_p})$ is the cross-entropy loss.

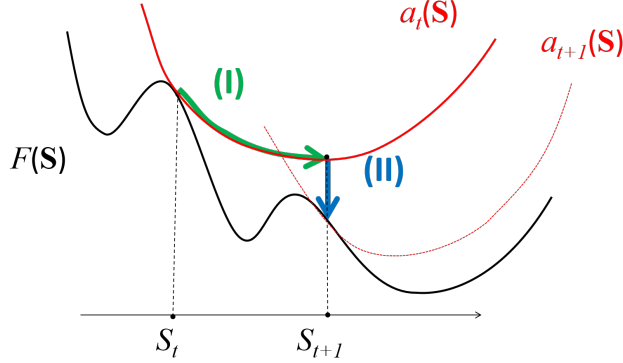


Figure 1.10: *Illustration of the general bound optimization procedure:* Iteration t of optimizing function $F(S)$ using auxiliary functions (bounds) $a_t(S)$. *Step I* minimizes $a_t(S)$. *Step II* computes the next bound $a_{t+1}(S)$.

Nowadays, almost all segmentation networks are fully convolutional, and lots of architectures have been proposed. These networks differ in the ways low-level features are aggregated with high-level features [45, 250, 46, 259, 257], i.e., combining fine details with semantics and context cues. Strided convolution and pooling layers reduce spatial resolution. Various ways of upsampling and variants of convolution [150, 45, 250] are brought forward to restore to the original spatial resolution. Ever-deeper networks are trained using ever-larger dataset, improving segmentation accuracy.

1.4 Our Motivation and Contribution

Having reviewed standard and relevant techniques for segmentation in Sec. 1.3, here we give a glimpse of our motivation, while elaboration is left in subsequent chapters. We identified limitations with these techniques. We have three important motivations discussed in Sec. 1.4.1 - 1.4.3. With these motivations and insights, we propose new formulations and corresponding efficient optimization methods that take advantage of the complementary benefits of different segmentation techniques.

1.4.1 Bound Optimization

Our first observation and motivation is that many segmentation technique such as K-means (1.1) is related to *bound optimization*. In general, bound optimizers are iterative algorithms

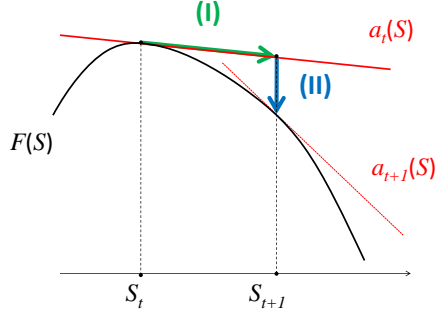


Figure 1.11: *K-means as linear bound optimization*: As obvious from (1.1), the bound $a_t(S)$ in Theorem 1 is a unary function of S . KM procedures (1.18) correspond to optimization of linear auxiliary functions $a_t(S)$ for KM objectives.

that optimize *auxiliary functions* (upper bounds) for a given objective $F(S)$ assuming that these auxiliary functions are more tractable than the original difficult optimization problem [136, 168, 18, 217].

Definition 4 (Auxiliary function). *Let t be a current iteration index, $a_t(S)$ is an auxiliary function of $F(S)$ at current solution S_t if*

$$F(S) \leq a_t(S) \quad \forall S \quad (1.14a)$$

$$F(S_t) = a_t(S_t). \quad (1.14b)$$

The auxiliary function is minimized at each iteration t (Fig. 1.10)

$$S_{t+1} = \arg \min_S a_t(S). \quad (1.15)$$

This procedure iteratively decreases function $F(S)$ since

$$F(S_{t+1}) \leq a_t(S_{t+1}) \leq a_t(S_t) = F(S_t).$$

Here, we show how K-means (KM) is related to bound optimization. The most basic iterative KM algorithm [69] can be described as the *block-coordinate descent* for the *mixed* objective $F(S, \mathbf{m})$ (1.1) combining discrete variables $S = \{S_p\}$ with continuous variables $\mathbf{m} = \{m_k\}_{k=1}^K$ representing cluster “centers”. For any given S the optimal centers

$\arg \min_{\mathbf{m}} F(S, \mathbf{m})$ are the means

$$\mu_{S^k} := \frac{\sum_{p \in S^k} I_p}{|S^k|} \quad (1.16)$$

where $S^k = \{p | S_p = k\}$ is the k^{th} segment and $|S^k|$ is the segment's cardinality. This reduce (1.1) to KM energy of the single variable S ,

$$F_{km}(S) = \sum_k \sum_{p \in S^k} \|I_p - \mu_{S^k}\|^2. \quad (1.17)$$

Assuming current segments S_t^k the update operation giving $\arg \min_S F(S, \mu_{S_t})$

$$\left(\begin{array}{l} \text{basic KM} \\ \text{procedure} \end{array} \right) \quad S_p \leftarrow \arg \min_k \|I_p - \mu_{S_t^k}\| \quad (1.18)$$

defines the next solution S_{t+1} as per standard K-means algorithm. This greedy descent technique converges only to a local minimum of KM objective (1.1), which is known to be NP hard to optimize. There are also other approximation methods.

We show that standard KM procedures correspond to bound optimization for K-means objective (1.1). Note that variables m_k in mixed objective $F_{km}(S, \mathbf{m})$ (1.1) can be seen as relaxations of segment means μ_{S^k} in single-variable KM objective $F_{km}(S)$ (1.17) since

$$\begin{aligned} \mu_{S^k} &= \arg \min_{m_k} \sum_{p \in S^k} \|I_p - m_k\|^2 \\ \text{and } F_{km}(S) &= \min_{\mathbf{m}} F_{km}(S, \mathbf{m}). \end{aligned} \quad (1.19)$$

Theorem 1 (bound for KM). *Standard iterative K-means procedure (1.18) is a bound optimization method for K-means objectives $F_{km}(S)$ (1.17) using auxiliary function*

$$a_t(S) = F_{km}(S, \mu_t) \quad (1.20)$$

at any current segmentation $S_t = \{S_t^k\}$ with means $\mu_t = \{\mu_{S_t^k}\}$.

Proof. Equation (1.19) implies $a_t(S) \geq F_{km}(S)$. Since $a_t(S_t) = F_{km}(S_t)$ then $a_t(S)$ is an auxiliary function for $F(S)$. Re-segmentation step (1.18) gives optimal segments S_{t+1} minimizing the bound $a_t(S)$. The re-centering step minimizing $F_{km}(S_{t+1}, m)$ for fixed segments gives means μ_{t+1} defining bound $a_{t+1}(S)$ for the next iteration. These re-segmentation (I) and re-centering (II) steps are illustrated in Figs. 1.10, 1.11. \square

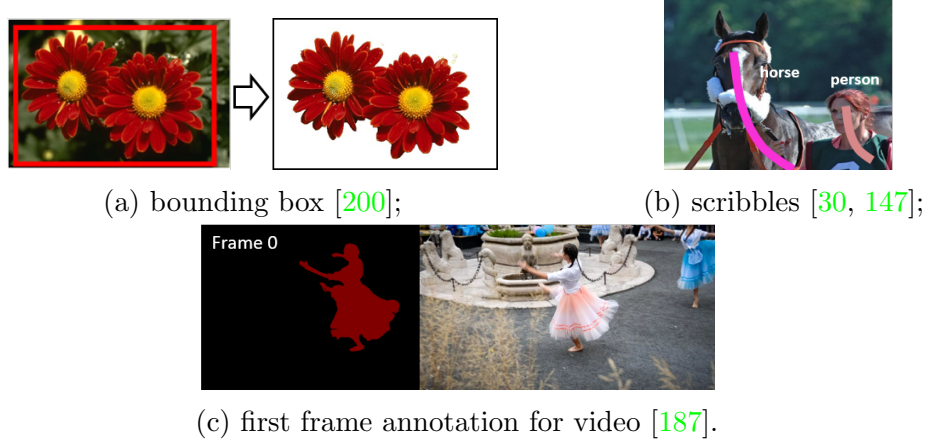


Figure 1.12: Weakly-supervised segmentation with different forms of weak supervision.

1.4.2 Weakly-supervised Segmentation and Semi-supervised Learning (SSL)

Often in segmentation tasks, we are given only some weak-supervision. For example, the users provide bounding boxes or strokes for interactive segmentation [30, 200]. For video and 3D volume in medical imaging, it is laborious to segment all frames in a video, so only the first frames are labeled [187]. Fig. 1.12 shows examples of weak supervision. Other types of supervision include for instance clicks [13], polygons [38], extreme point [156], and image-level tags [121]. Training a CNN for segmentation with weak-supervision has attracted lots of interests [147, 13, 121, 156].

We address weakly-supervised segmentation by principled approaches from semi-supervised learning [265, 44], which is about learning from both labeled and unlabeled data [265, 44]. Fig. 1.13 shows a toy example of semi-supervised classification.

Definition 5 (Semi-supervised Learning). *Given M labeled data points $(x_i, y_i) \in (\mathcal{X}, \mathcal{Y})$, $i = 1, \dots, M$ and U unlabeled data points $x_i \in \mathcal{X}$, $i = M + 1, \dots, M + U$, learn $f(x) : \mathcal{X} \rightarrow \mathcal{Y}$.*

Weakly-supervised segmentation and semi-supervised learning are closely related. For example, segmentation with scribbles is essentially a SSL problem since scribbled pixels are labeled while the rest are unlabeled. In fact, standard MRF formulation for interactive segmentation [30] is very similar to SSL formulation based on graph regularization [264, 265]. However, they are independently developed in computer vision and machine learning

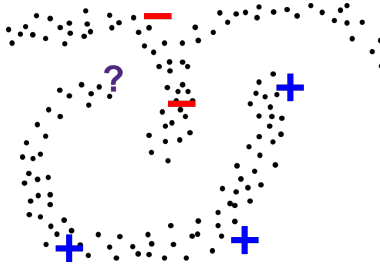


Figure 1.13: A synthetic example of semi-supervised learning with labeled and unlabeled data for binary classification. Given unlabeled data, the test point (?) is expected to be of positive class though it is closer to negative samples.

communities. We revisit the connections and leverage well-established SSL techniques for weakly-supervised CNN segmentation, which is dominated by heuristically training from fully labeled proposals [147, 121, 116].

Semi-supervised learning works since certain assumptions hold [44]. For example, it is natural to assume that two points x_1, x_2 nearby in feature space \mathcal{X} should be of the same output label y . Also, the decision boundary should lie in the low-density region.

Semi-supervised learning algorithms include for example self-training, generative models, co-training, semi-supervised support vector machines, and graph-based algorithms. Simple self-learning [79], as a wrapper method, is sensitive to the reinforcement of early mistakes. The iterative EM algorithm maximizes the likelihood of labeled and unlabeled data with generative models [175], e.g. Gaussian Mixture Models (GMMs). Transductive SVM (TSVM) maximizes the margin for both labeled and unlabeled data, but its optimization is difficult [44]. Graph-based algorithms [24, 264, 262, 15] are the most relevant to our work, which assume that two nodes with larger graph affinity are more likely to have the same label. Graph Cut [24] solves a combinatorial problem in polynomial time. Harmonic Functions [264] relaxes discrete labeling to real values and admits a closed-form solution. Manifold regularization [15] prevents over-fitting to training examples by including extra regularization on the whole feature space. As such, it gives better generalization and a natural extension to test data.

Deep learning needs lots of labeled data in general, which is particularly problematic for pixel-wise segmentation since such ground-truth is very expensive to obtain. We adapt principled SSL techniques to weakly-supervised CNN segmentation. Our proposed regularized loss framework is simple, general, and applicable to a wide range of weak-supervisions such as scribbles, clicks, and image-level tags.

1.4.3 Combining KC, MRF, and CNN

While CNNs [150, 45, 101] achieved record-breaking accuracy on benchmarks, segmentation is far from being solved. CNN segmentation has the following limitations.

1. Training needs lots of fully labeled images. Manual image segmentation is very tedious and time-consuming. For example, it takes approximately $10 \text{ min} \times 11,000 = 73 \text{ days}$ for PASCAL dataset [74] that contains 11,000 training images. Also, it takes much more efforts to annotate videos. For modality like medical images, expertise is required to segment the training images precisely.
2. Segmentation networks trained on one dataset often don't generalize well on another dataset of different-looking images. However, generalization is critical to enable deployment of computer vision systems for various scenes. An example is road segmentation in autonomous driving for different cities with different street views. It is too costly and infeasible to annotate data and retrain the networks for each city. Fully supervised training has limitation and is not scalable.
3. Output segmentation from CNNs may not align well with object boundaries, in particular for thin structured objects. Coarse upsampling from downsampled feature maps leads to poor boundary alignment. Post-processing by e.g. MRF [45] also improves segmentation on the boundary. However, few works explicitly incorporate boundary alignment and other regularization into training [260, 207, 11].
4. CNN segmentation does not model geometric constraints and priors about the object of interest. Ideally, networks are supposed to learn everything including geometry about objects. However, it is not the case in practice. For example, output segmentation can still be fragmented for objects that should be connected components [233]. Also, it is natural to have shape prior [57, 229, 92] for objects. However, there is no mechanism to constrain the shape of output segmentation.

Shallow v.s. Deep Segmentation: To address some of the issues with CNN segmentation, we propose to combine shallow and deep segmentation techniques. Shallow segmentation methods including MRF [30] and clustering [211, 64] are complementary to deep segmentation from the following reasons. Firstly, clustering or MRF techniques are unsupervised or weakly-supervised with a little amount of annotation, for example with scribbles or first-frame annotation for video segmentation. On the contrary, the mainstream of CNN segmentation is based on full-supervision on tens of thousands of images.

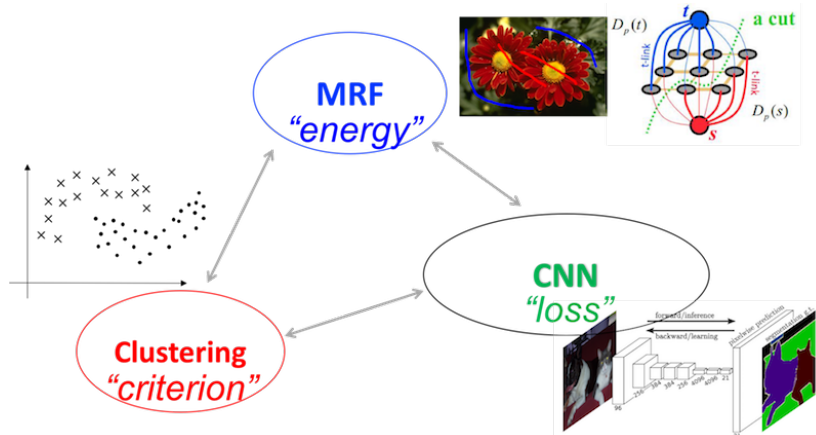


Figure 1.14: We study optimization in image segmentation with particular focus on combining these techniques in a principled way. For example, we jointly optimize MRF energy and clustering criterion in our Kernel Cut method [223]. Our regularized loss framework [224] bridges deep CNN segmentation and shallow segmentation methods including MRF and clustering.

Secondly, the formulation of shallow segmentation is usually motivated by *regularization*, *prior* and *constraints* including boundary alignment, color clustering, volume prior, shape prior, and geometric constraints. These regularizations are exactly what’s lacking for CNN segmentation. Thirdly, shallow segmentation is usually based on raw or low-level features including color and texture while deep segmentation is good at extracting discriminant higher-level features.

A naive way to combine shallow and deep segmentation is to extract features from CNNs for shallow segmentation techniques. This work goes beyond and we propose to combine shallow with deep segmentation in a principled way through minimization of *regularized loss*. Segmentation CNNs are typically trained to minimize pixel-wise classification loss w.r.t. target label, i.e., empirical risk loss. Shallow segmentation is often formulated as minimizing some objectives referred as MRF “energies” or clustering “criteria” in the literature. So we take these objectives as regularization part and combine with empirical risk loss as a regularized loss for CNN segmentation.

Fig. 1.14 gives a schematic overview of related segmentation techniques and the corresponding objectives to be optimized. In this thesis, we focus on the formulation and optimization of segmentation techniques including MRF, Clustering, and CNN.

1.5 Outline of the Thesis and Publication

In Chapter 2, we study optimization of MRF energies that are high-order or non-submodular. Minimization of such energies is generally NP-hard. We propose *pseudo-bound optimization* [217] that relaxes the upper-bound condition for an auxiliary function and to replace it with a family of pseudo-bounds. Our Pseudo-Bound Cuts algorithm improves the state-of-the-art in many applications: appearance entropy minimization, target distribution matching, and curvature regularization.

In Chapter 3, we combine MRF regularization with Clustering, which are two popular methodologies for image segmentation. We identify previously unknown secrets and bias of MRF based segmentation and kernel clustering [218, 158]. We explain how regularization and kernel clustering can work together and why this is useful. Our joint energy combines standard regularization, e.g. MRF potentials, and kernel clustering criteria like normalized cut. Complementarity of such terms is demonstrated in many applications using our bound optimization *Kernel Cut* algorithm for the joint energy.

In Chapter 4, we combine deep CNN segmentation with non-deep or shallow techniques including MRF and clustering. We propose MRF energies and kernel clustering criterion as regularization losses [219, 224] for weakly supervised CNN segmentation. Minimization of regularized losses is a principled approach to semi-supervised learning well-established in deep learning, in general. However, it is largely overlooked in semantic segmentation currently dominated by methods mimicking full supervision via "fake" fully-labeled masks (proposals) generated from available partial input. We propose different regularization losses including normalized cut [219], MRF [224], and volume constraints [115]. We also investigated novel optimization algorithm for regularized segmentation losses beyond gradient descent [157].

In Chapter 5, we conclude the thesis and discuss future work & new directions.

This thesis is based the following published articles.

1. "Pseudo-Bound Optimization for Binary Energies", **Meng Tang**, Ismail Ben Ayed, Yuri Boykov, In *European Conference on Computer Vision (ECCV)*, Zurich, Switzerland, September 2014.
2. "Secrets of GrabCut and Kernel K-means", **Meng Tang**, Ismail Ben Ayed, Dmitrii Marin, Yuri Boykov, In *IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, December 2015.

3. "Normalized Cut Meets MRF", **Meng Tang**, Dmitrii Marin, Ismail Ben Ayed, Yuri Boykov, In *European Conference on Computer Vision (ECCV)*, Amsterdam, the Netherlands, October 2016.
4. "Kernel Clustering: Density Biases and Solutions", Dmitrii Marin, **Meng Tang**, Ismail Ben Ayed, Yuri Boykov, In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2017.
5. "Normalized Cut Loss for Weakly-supervised CNN Segmentation", **Meng Tang**, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, Christopher Schroers, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, USA, June 2018.
6. "On Regularized Losses for Weakly-supervised CNN Segmentation", **Meng Tang**, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, Yuri Boykov, In *European Conference on Computer Vision (ECCV)*, Munich, Germany, September 2018.
7. "Kernel Cuts: Kernel and Spectral Clustering meet Regularization", **Meng Tang**, Dmitrii Marin, Ismail Ben Ayed, Yuri Boykov, In *International Journal of Computer Vision (IJCV)*, 2019.
8. "Beyond Gradient Descent for Regularized Segmentation Losses", Dmitrii Marin, **Meng Tang**, Ismail Ben Ayed, Yuri Boykov, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, USA, June 2019.
9. "Constrained-CNN losses for weakly supervised segmentation", Hoel Kervadec, Jose Dolz, **Meng Tang**, Eric Granger, Yuri Boykov, Ismail Ben Ayed, In *Medical Image Analysis (MedIA)*, Volume 54, P. 88-99, May 2019.

Chapter 2

Pseudo-bound Optimization for Binary MRF Energies

2.1 Introduction

Recently high-order [17, 18, 91, 95, 112, 164, 188, 243] and non-submodular pairwise [90, 113, 126, 123] energy minimization have drawn tremendous research interests. Those energy functions arise naturally in many computer vision and image processing applications. Examples of high-order functions include but are not limited to constraints on segment volume [91, 243], clique labeling consistency [95, 120, 220] and matching target distributions [17, 18, 91, 188]. Pairwise non-submodular energies occur in deconvolution [90], curvature regularization [72, 90, 173], inpainting [113] and surface registration [113].

In general, optimization of high-order or non-submodular pairwise energy is *NP-hard*. Existing approximation methods make optimization tractable either by *global* or *local* linearization. Well established LP relaxation methods such as QPBO [25, 198] and TRWS [123] are examples of global linearization techniques for solving non-submodular energies in vision [113]. By relaxing the integrality constraints, they globally transform the original function into a linear function with extra variables and linear constraints. Unlike global linearization, local techniques iteratively approximate the original energy around current solution, for instance, using Taylor approximations [90, 91, 117, 143] or auxiliary functions [17, 18, 90, 168, 188, 199]. The recent Fast Trust Region (FTR) method [91] finds the optimal solution of a local approximation within a trust region, *i.e.* a region near current solution where the approximation can be trusted. The trust region size is adaptively adjusted depending on the quality of current approximation using well-known

trust region paradigms [255]. The recent studies [90, 91] have shown that FTR achieved the state of the art performance in many applications. Our work is more closely related to bound optimizers, which take auxiliary functions [17, 18, 90, 136, 168, 188, 199] as local approximations and were recently shown to yield competitive performances in several vision problems [18, 90, 188].

In this paper, we tackle binary energy functions $E(S)$ where $S = (S_p | p \in \Omega) \in \{0, 1\}^{|\Omega|}$ is a vector of binary variables. We let bold font $\mathbf{S} = \{p | S_p = 1\} \subset \Omega$ be the set of pixels labeled as foreground, and $\bar{\mathbf{S}} = \Omega \setminus \mathbf{S}$ as the complementary background segment.

2.1.1 Bound optimization

Bound optimizers iteratively minimize an *auxiliary function* bounding the original energy across the entire solution space, see its definition 4. Denote the auxiliary function to be $A_t(S)$, then the current solution S_t is updated to the global optimum of the auxiliary function:

$$S_{t+1} = \arg \min_S A_t(S), \quad t = 1, 2, \dots \quad (2.1)$$

Ideally optimization of the auxiliary function is easier than that of the original energy. Bound optimizers guarantee not to increase the original energy at each iteration,

$$E(S_{t+1}) \leq A_t(S_{t+1}) \leq A_t(S_t) = E(S_t). \quad (2.2)$$

Examples of well-known bound optimizers include mean-shift [75], difference of convex functions (DC) programming techniques [5], expectation maximization (EM) and submodular-supermodular procedures [168]. Besides, bound optimizers successfully tackled various problems in machine learning [258], computational statistics [136] and nonnegative matrix factorization [139]. In vision, bound optimizers were recently used for high-order or non-submodular pairwise energies [18, 188, 90, 199]. The recent Auxiliary Cuts [18] work derived bounds for certain class of high-order functions. A variant of Auxiliary Cuts (LSA-AUX) is proposed in [90] for quadratic pseudo-boolean optimization.

2.1.2 Motivation and Contributions

Typically, for bound optimizers, one auxiliary function is chosen and optimized at each iteration. Furthermore, such an auxiliary function has to be an upper bound for the original energy $E(S)$ across the entire solution space, see $A_t(S)$ in Fig.2.1. However, in

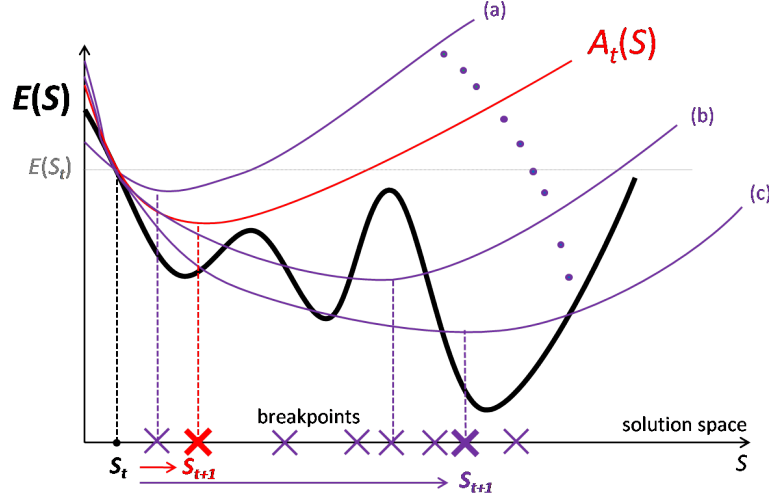


Figure 2.1: Illustration of how pseudo-bound optimization framework updates current solution from S_t to S_{t+1} . Instead of optimizing only one auxiliary function $A_t(S)$, we explore a parametric family of pseudo-bounds. The best solution with the minimum original energy is chosen among the set of all global minima for the family. They correspond to breakpoints of parametric maxflow method [125] that can be efficiently explored in polynomial time. By optimizing the family of pseudo-bounds, larger decrease in energy is achieved ($S_t \rightarrow S_{t+1}$). [Best viewed in color]

practice, it is difficult to find bounds that approximate well the original energy while being amenable to fast global solvers. Although working well for some applications, *auxiliary cuts* [18, 90] may converge to undesirable solutions for several types of functions, see a representative example in Fig. 2.2.

Our main idea is to relax the bound condition for an auxiliary function replacing it with pseudo-bounds, which may better approximate the original energy. Consider the example in Fig. 2.1. Auxiliary function $A_t(S)$ does guarantee that its global minimum decreases the original energy $E(S)$, see Sec. 2.1.1. However, there are many other approximation functions whose global minimum also decrease the original energy. For example, optimal solutions for (a) and (c) decrease $E(S)$ because these functions are local upper bounds for $E(S)$ around their global minima. Function (b) does not have this local bound property, but its global minimum still decreases the original energy. Moreover, solutions obtained by minimizing other approximation functions, e.g. (c), could be better than the one from the upper bound, *i.e.* auxiliary function $A_t(S)$. In that sense, relaxing the upper bound constraint allows better approximations of $E(S)$.

We want to design an optimization algorithm using a larger class of relaxed bounds, which could give better solutions than proper auxiliary functions. The key challenge is choosing such *pseudo-bounds* so as to guarantee the original energy decrease; note that the upper bound constraint was used when proving (2.2). One way to proceed could be to design some specific relaxed bounds that guarantee the decrease for $E(S)$ by construction. For example, in some applications it might be possible to design a particular approximation function that is guaranteed to locally dominate the original energy only around its global optimum, as in Fig.2.1 (a) or (c), which is sufficient to prove the decrease of $E(S)$.

This paper follows an alternative approximation approach. Instead of a single auxiliary function, at each iteration we optimize a family of *pseudo-bounds* that includes only one proper bound, while the bound constraint is relaxed for the other functions. As shown in Sec.2.2.1, inclusion of one proper bound is sufficient to guarantee the original energy decrease when the best solution is selected among global minima for the whole family. As illustrated in Fig.2.1 and confirmed by our experiments, relaxation of the bound constraint allows to significantly improve the quality of optimization compared to auxiliary functions, even when pseudo-bounds come from the same class of globally optimizable functionals.

A parametric family of pseudo-bounds is built as follows. We start from a known optimizable, *i.e.* submodular, auxiliary function and add a unary *bound relaxation* term weighted by a parameter. In order to explore all global minima for the whole parametric family efficiently, we propose parametric maxflow [125, 105, 81], reviewed in Sec. 2.2.2. To find all global minima for the whole family in polynomial time, the unary bound relaxation term must be monotone w.r.t. parameter. This practical consideration is important when selecting parametric pseudo-bound families for specific applications, e.g appearance entropy (2.11), distribution matching (2.15), or supermodular function (2.18). Note that parametric maxflow can be easily parallelized.

Our contributions can be summarized as follows.

- This paper proposes a new general *pseudo-bound optimization* paradigm for approximate iterative minimization of high-order and non-submodular binary energies. It is a generalization of the standard *majorize-minimize* principle relaxing the bound constraint for an auxiliary function.
- We optimize a parametric family of pseudo-bounds at each iteration. To guarantee the energy decreases we include one proper bound in the family.
- In the context of discrete optimization, we propose parametric maxflow technique [125, 105, 81] to explore all global minima for the whole family in low-order polyno-

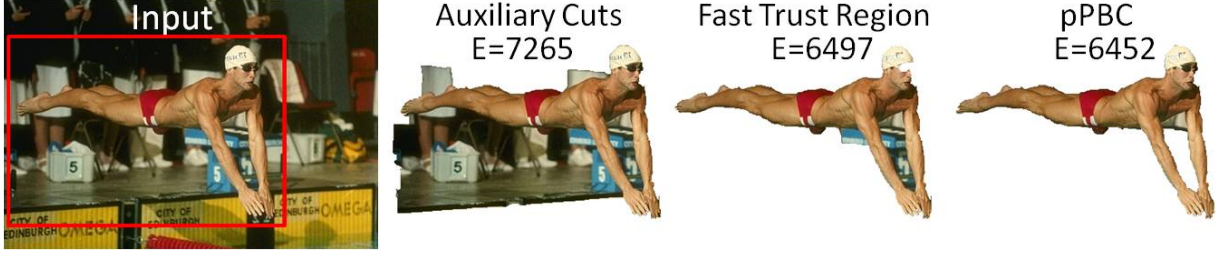


Figure 2.2: Matching target foreground color distribution using auxiliary cuts [18], fast trust region [91] and pPBC. pPBC achieves the lowest energy.

mial time. To guarantee this complexity, we can choose families of pseudo-bounds with *monotone* dependence on parameter.

- We propose and discuss several examples of pseudo-bound families for different high-order and non-submodular pairwise energies.
- Our *parametric Pseudo-Bound Cuts* algorithm (pPBC) improves the-state-of-the-art in many energy minimization problems, e.g. entropy based image segmentation, target distributions matching, curvature regularization and image deconvolution. In particular, we outperform the standard GrabCut algorithm [200] both in terms of energy and segmentation error statistics. Our pseudo-bound approach is more robust to initialization and binning. Our pPBC algorithm also gives lower energy than *Auxiliary Cuts* [18] and *Fast Trust Region* [91] for distribution matching, see Fig. 2.2, and other challenging optimization problems in computer vision, see Section 3.5.

2.2 Parametric Pseudo-Bound Cuts (pPBC)

2.2.1 Our pseudo-bound framework

First, we define a family of *pseudo-bounds* for a scalar parameter λ with values in some set $\Lambda \subseteq \mathbb{R}$, for example $\Lambda = [\lambda^{\min}, \lambda^{\max}]$.

Definition 6 (pseudo-bound). Assume energy $E(S)$, some current solution $S_t \in \{0, 1\}^\Omega$ and parameter $\lambda \in \Lambda$. Then, function $\mathcal{F}_t(S, \lambda) : \{0, 1\}^\Omega \times \Lambda \rightarrow \mathbb{R}$ is called a pseudo bound for energy $E(S)$ if there exists $\lambda' \in \Lambda$ such that $\mathcal{F}_t(S, \lambda')$ is an auxiliary function for $E(S)$ at current solution S_t .

Algorithm 2: PARAMETRIC PSEUDO-BOUND CUTS (PPBC)

```

1  $S_0 \leftarrow S_{init}$ ;
2 For  $t = 0, 1, 2, \dots$ , repeat until convergence ;
3   Construct an auxiliary function  $A_t(S)$  at current solution  $S_t$ ;
4   Combine  $A_t(S)$  with unary relaxation term  $R_t(S)$  to form pseudo-bound
       $\mathcal{F}_t(S, \lambda) = A_t(S) + \lambda R_t(S)$ ;
5   //Optimize the parametric family of pseudo-bounds
       $S^\lambda = \arg \min_S \mathcal{F}_t(S, \lambda)$ , for  $\lambda \in \Lambda$ ;
6   //Score candidate solutions and update
       $\lambda^* = \arg \min_\lambda E(S^\lambda)$ ,  $S_{t+1} \leftarrow S^{\lambda^*}$ ;

```

We may informally refer to pseudo-bound function $\mathcal{F}_t(S, \lambda)$ as a family of pseudo-bounds or a parametric family.

Our goal is to iteratively update current solution S_t for energy $E(S)$. Instead of bound optimization discussed in Sec. 2.1.1, we propose to compute new better solution S_{t+1} by optimizing pseudo-bound $\mathcal{F}_t(S, \lambda)$ as follows.

Proposition 1. *Assume energy $E(S)$, current solution S_t and a pseudo-bound family $\mathcal{F}_t(S, \lambda)$ over parameter $\lambda \in \Lambda$. Let S^λ denote an optimal solution for $\mathcal{F}_t(S, \lambda)$ at any particular λ :*

$$S^\lambda = \arg \min_S \mathcal{F}_t(S, \lambda). \quad (2.3)$$

Then, $\lambda^ = \arg \min_\lambda E(S^\lambda)$ gives solution $S_{t+1} := S^{\lambda^*}$ reducing original energy*

$$E(S_{t+1}) = E(S^{\lambda^*}) \leq E(S_t).$$

Proof. Pseudo-bound family $\mathcal{F}_t(S, \lambda)$ contains an auxiliary function $\mathcal{F}_t(S, \lambda')$ for some λ' . Optimization over the whole family should give better solution than one particular function $E(S^{\lambda^*}) \leq E(S^{\lambda'})$. Then, the proposition follows from the property of auxiliary functions $E(S^{\lambda'}) \leq E(S_t)$, see (2.2). \square

We construct a pseudo-bound family at current solution S_t by augmenting some auxiliary function $A_t(S)$ with a weighted *bound relaxation* term $R_t(S)$:

$$\mathcal{F}_t(S, \lambda) = A_t(S) + \lambda R_t(S). \quad (2.4)$$

Note that for $\lambda = 0$ our pseudo-bound $\mathcal{F}_t(S, \lambda)$ in (2.4) reduces to auxiliary function $A_t(S)$. Starting from the same current solution S_t , our pseudo-bound optimization is guaranteed

to find at least as good or better solution than optimization of bound $A_t(S)$. While pseudo-bound may not be a proper bound for $\lambda \neq 0$, it may better approximate the original energy $E(S)$, see Fig.2.1.

In the context of binary energies $E(S)$ we typically choose some submodular $A_t(S)$ and modular (unary) $R_t(S)$. The resulting pseudo-bound family (2.4) is of the form (2.5) that allows to efficiently explore the whole set of solutions S^λ with standard *parametric maxflow* techniques reviewed in Sec. 2.2.2. The next solution $S_{t+1} = S^{\lambda^*}$ can be computed by selecting S^λ with the lowest value of original energy $E(S)$, as summarized in Alg.2.

2.2.2 Overview of *parametric maxflow*

Parametric maxflow technique [125, 105, 81] is a building block in our proposed algorithm. For all λ in some interval $\Lambda = [\lambda^{min}, \lambda^{max}]$, parametric maxflow can efficiently generate a (finite) set of all distinct solutions $S^\lambda \in \{0, 1\}^\Omega$ minimizing energy $E(S, \lambda)$ of form

$$S^\lambda = \arg \min_S \overbrace{\sum_{p \in \Omega} (a_p + \lambda b_p) s_p + \sum_{(p,q) \in \mathcal{N}} \phi_{pq}(s_p, s_q)}^{E(S, \lambda)} \quad (2.5)$$

where ϕ_{pq} are submodular pairwise terms for a set of pairwise factors \mathcal{N} . Note that the unary terms in (2.5) *linearly* depend on parameter λ .

As discussed in [71, 125], interval Λ can be broken into a finite set of subintervals between *breakpoints* $\lambda_1 < \lambda_2 < \dots < \lambda_k \in \Lambda$ such that any λ inside each given interval $[\lambda_i, \lambda_{i+1}]$ gives the same solution $S^\lambda = S^i$. Parametric maxflow identifies all breakpoints and solutions S^i by making a finite number of calls to the maxflow procedure, see [125, 71] for details. Importantly, in *monotonic* case when coefficients b_p in (2.5) are either all non-negative or all non-positive, optimal solutions S^i have a *nestedness* property leading to guaranteed *polynomial* complexity. This necessitates our choice of relaxation term $R_t(S)$ to have unary coefficients of the same sign.

2.3 Examples of pseudo-bounds

Algorithm 2 for minimizing energy $E(S)$ depends on pseudo-bound (2.4) and requires specific choices of a submodular auxiliary function $A_t(S)$ and a unary relaxation term $R_t(S)$. This section provides practical pseudo-bound examples for a wide range of high-order and non-submodular pairwise energies $E(S)$.

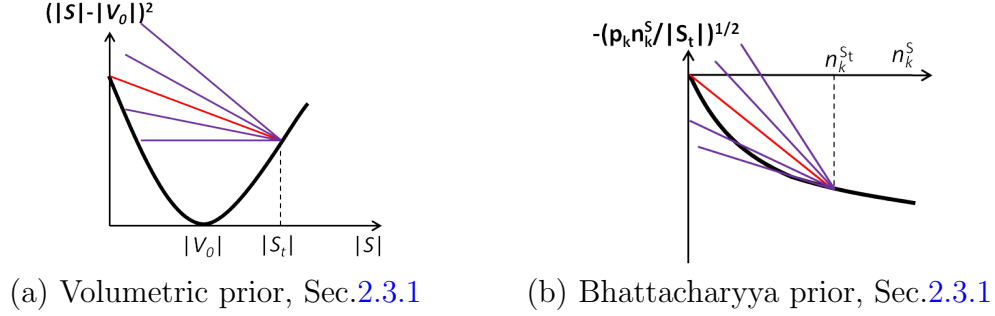


Figure 2.3: Pseudo-bound families for two cardinality functions. Auxiliary functions are red.

2.3.1 High-order energies

Volumetric potential [91, 243]

Energy like $\psi(|\mathbf{S}| - V_0)$ for convex symmetric function $\psi(\cdot)$ penalizes deviation of size of segment $|\mathbf{S}|$ from target volume V_0 . For example, if $\psi(x) = x^2$ and $\mathbf{S} \subset \mathbf{S}_t$ we can use the following pseudo bound family¹ illustrated in Fig.2.3(a)

$$\mathcal{F}_t(S, \lambda) = \underbrace{(|\mathbf{S}_t| - 2V_0)|\mathbf{S}| + V_0^2}_{A_t(S)} + \lambda \underbrace{(|\mathbf{S}| - |\mathbf{S}_t|)}_{R_t(S)}. \quad (2.6)$$

Appearance Entropy

Appearance entropy was proposed for image segmentation in [220] as a general color consistency criterion. It can be combined with other standard terms, e.g. boundary smoothness, as in the following binary segmentation energy known as maximum description length (MDL)

$$E_{MDL}(S) = |\mathbf{S}| \cdot H(\mathbf{S}) + |\bar{\mathbf{S}}| \cdot H(\bar{\mathbf{S}}) + |\partial S| \quad (2.7)$$

where $H(\mathbf{S})$ and $H(\bar{\mathbf{S}})$ are entropies of color histograms inside foreground \mathbf{S} and background $\bar{\mathbf{S}}$ and $|\partial S| = \sum_{\{p,q\} \in \mathcal{N}} \omega_{pq} |s_p - s_q|$ is segmentation boundary length. Indirectly, color entropy was also used for segmentation in [263, 200, 60]. Entropy can also be used as a clustering criterion for any image features that can be binned. In fact, entropy and related

¹For $\psi(x) = x^2$ our volumetric potential is non-submodular pairwise, see also Sec.2.3.2.

information gain criterion are widely used in learning, e.g. for contextual clustering [145] or decision trees [212].

Entropy-based energy (2.7) from [220] is related to a well-known *minimum description length* (MDL) functional [263, 60] for color model fitting. In particular, for two segments it reduces to a color model fitting energy in GrabCut [200]

$$E_{MDL}(S, \theta^1, \theta^0) = \sum_{p \in \Omega} -\log P(I_p | \theta^{s_p}) + |\partial S| \quad (2.8)$$

where θ^1 and θ^0 are variables corresponding to unknown color models for foreground \mathbf{S} and background $\bar{\mathbf{S}}$. As shown in [220], globally optimal S for high-order energy (2.7) and mixed optimization functional (2.8) coincide if color models θ are represented by histograms. Since (2.8) is known to be *NP-hard* [234], it follows that high-order entropy energy in (2.7) is also *NP-hard*.

Equivalence of global solutions for entropy (2.7) and color-model fitting (2.8) suggests that (2.7) is minimized indirectly when applying standard *block-coordinate descent* (BCD) techniques [263, 200, 60] to energy (2.8) separately optimizing variables S and θ at each iteration. Below, we show that BCD in [200] can be seen as a bound optimization method for entropy (2.7). Then, we use the corresponding auxiliary function to build a family of pseudo-bounds that generate significantly better results, as shown by our experiments in Sec.2.4.1.

Proposition 2. Assume fixed histograms θ_t^1 and θ_t^0 computed from the colors of current solution S_t (foreground) and its complement \bar{S}_t (background). Then,

$$A_t(S) := E_{MDL}(S, \theta_t^1, \theta_t^0), \quad (2.9)$$

with E as in (2.8), is an auxiliary function for entropy-based energy (2.7) at S_t .

Proof. It follows from a cross entropy discussion in [220]. Indeed, it is easy to check

$$E_{MDL}(S, \theta_t^1, \theta_t^0) = |\mathbf{S}| \cdot H(\mathbf{S} | \mathbf{S}_t) + |\bar{\mathbf{S}}| \cdot H(\bar{\mathbf{S}} | \bar{\mathbf{S}}_t) + |\partial S| \quad (2.10)$$

where $H(\cdot | \cdot)$ is a cross-entropy of color distributions in two sets of pixels. Inequality $H(\mathbf{A} | \mathbf{B}) \geq H(\mathbf{A} | \mathbf{A}) = H(\mathbf{A}) \ \forall \mathbf{A}, \mathbf{B} \subset \Omega$ implies

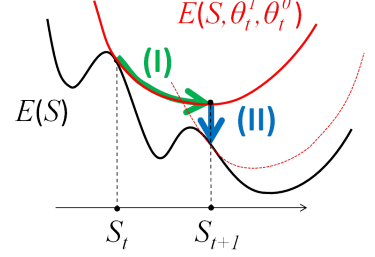
$$E_{MDL}(S, \theta_t^1, \theta_t^0) \geq E_{MDL}(S)$$

where $E_{MDL}(S)$ is from (2.7). It is also easy to check that $E_{MDL}(S_t, \theta_t^1, \theta_t^0) = E_{MDL}(S_t)$. \square

Corollary 1. Block-coordinate descent (BCD) for mixed functional (2.8), as in GrabCut [200], is a bound optimization for entropy-based energy (2.7), see Sec.2.1.1.

Proof. Two steps during each iteration of BCD in GrabCut are (I) optimization of

segmentation S by applying graph cuts to energy (2.8) with fixed color models, as in Boykov-Jolly [30], and (II) optimization of color models θ^0, θ^1 in energy (2.8) with fixed segmentation. Prop. 2 implies that the segmentation step optimizes auxiliary function $A_t(S)$ for energy (2.7) at S_t and gives next solution S_{t+1} , as illustrated on the right. Color model re-estimation step gives new auxiliary function $A_{t+1}(S)$ at S_{t+1} .



Our proposed pPBC method for entropy-based segmentation energy (2.7) augments the auxiliary function $A_t(S)$ in (2.9) with weighted bound relaxation term $\lambda(|\mathbf{S}| - |\mathbf{S}_t|)$ giving the following family of pseudo-bounds:

$$\mathcal{F}_t(S, \lambda) = E_{MDL}(S, \theta_t^1, \theta_t^0) + \lambda(|\mathbf{S}| - |\mathbf{S}_t|). \quad (2.11)$$

Matching color distributions [18, 91, 188, 17]

One way of matching target color distributions is to minimize Bhattacharyya measure:

$$Bha(S | p) = - \sum_k \sqrt{p_k n_k^{\mathbf{S}} / |\mathbf{S}|}, \quad (2.12)$$

where $n_k^{\mathbf{S}}$ is the number of foreground pixels in color bin k and $\sum_k p_k = 1$ is the target distribution. For $\mathbf{S} \subset \mathbf{S}_t$, a family of pseudo-bounds (Fig. 2.3) is given as:

$$\mathcal{F}_t(S, \lambda) = - \underbrace{\sum_k \sqrt{\frac{p_k}{n_k^{\mathbf{S}_t} |\mathbf{S}_t|}} n_k^{\mathbf{S}}}_{A_t(S)} + \lambda \underbrace{(|\mathbf{S}| - |\mathbf{S}_t|)}_{R_t(S)} \quad (2.13)$$

Another option for matching distributions is to use the KL divergence [18]:

$$KL(S) = \sum_k p_k \log \frac{p_k}{n_k^{\mathbf{S}} / |\mathbf{S}| + \epsilon} = \sum_k p_k \log p_k - \sum_k p_k \log \left(\frac{n_k^{\mathbf{S}}}{|\mathbf{S}|} + \epsilon \right), \quad (2.14)$$

where ϵ is a small constant used to avoid numerical issue. In this case, for $\mathbf{S} \subset \mathbf{S}_t$, we have the following family of pseudo-bounds (omitting constant $\sum_k p_k \log p_k$):

$$\mathcal{F}_t(S, \lambda) = \underbrace{\sum_k \frac{p_k}{n_k^{\mathbf{S}_t}} \left(\log \frac{\epsilon}{\frac{n_k^{\mathbf{S}_t}}{|\mathbf{S}_t|} + \epsilon} \right) n_k^{\mathbf{S}} - \log \epsilon}_{A_t(S)} + \lambda \underbrace{(|\mathbf{S}| - |\mathbf{S}_t|)}_{R_t(S)}, \quad (2.15)$$

where $A_t(S)$ is the auxiliary function derived recently in [18].

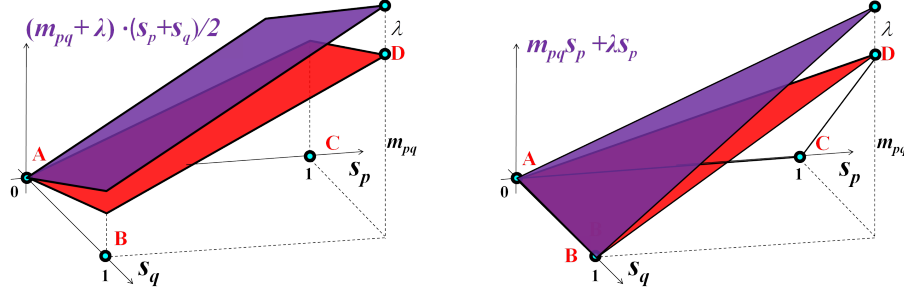


Figure 2.4: **pPBC-T**: Pseudo-bounds (purple) and auxiliary functions (red) of non-submodular potential $m_{pq}s_p s_q$ for current configuration $s_{p,t} = 0, s_{q,t} = 0$ (left) and $s_{p,t} = 0, s_{q,t} = 1$ (right).

2.3.2 Non-submodular pairwise energies

We consider a general class of binary pairwise non-submodular energies, which are useful in various vision applications [113, 90], e.g., segmentation, stereo, inpainting, deconvolution, and many others. Such energies can be expressed as:

$$E(S) = \sum_{(p,q) \in \mathcal{N}} m_{pq} S_p S_q = S^T M S, \quad S \in (0, 1)^\Omega \quad (2.16)$$

where $M = \{m_{pq} \in \mathbb{R} \mid p, q \in \Omega\}$ is a symmetric matrix containing pairwise potentials. if $m_{pq} \leq 0 \forall (p, q)$, energy (2.16) is *submodular* and, therefore, global optima can be reached in a low-order polynomial time using graph cuts [25]. The general non-submodular case is *NP-hard*. In the following, we propose three different pseudo-bounds families for (2.16) for non-submodular pairs ($m_{pq} > 0$).

pPBC-T(touch) gives pseudo-bounds for each non-submodular potential $m_{pq} S_p S_q$, $m_{pq} > 0$. Depending on the current configuration $S_{p,t}$ and $S_{q,t}$ for S_p and S_q , we augment the bound recently proposed in [90] with the relaxation terms specified as in Table 2.1. Fig. 2.4 shows the auxiliary functions in red and pseudo-bounds in purple for current configuration (0,0) and (0,1). Note that the bound relaxation term for current configuration (0,1) and (1,0) is different from that of (0,0) and (1,1). This relaxation allows the pseudo-bounds to *touch* the original energy at as many points as possible, yielding better approximation.

pPBC-B(ballooning) This option uses the auxiliary function in Table 2.1 augmented with a linear ballooning term $\lambda (|S| - |S_t|)$.

pPBC-L(Laplacian) We derive the third pseudo-bounds family based on the *Laplacian* matrix. Let $d(p) = \sum_q m_{pq}$ and D be the diagonal matrix having d on its diagonal. Notice

$(S_{p,t}, S_{q,t})$	Auxiliary function	relaxation term (pPBC-T)
$(0, 0)$	$m_{pq}(S_p + S_q)/2$	$\lambda(S_p - S_{p,t} + S_q - S_{q,t})$
$(0, 1)$	$m_{pq}S_p$	$\lambda(S_p - S_{p,t})$
$(1, 0)$	$m_{pq}S_q$	$\lambda(S_q - S_{q,t})$
$(1, 1)$	$m_{pq}(S_p + S_q)/2$	$\lambda(S_p - S_{p,t} + S_q - S_{q,t})$

Table 2.1: Auxiliary functions [90] and weighted bound relaxation term for pPBC-T.

that, in the case of supermodular terms ($m_{pq} \geq 0$), D is diagonally positive and, therefore, positive semidefinite. With symmetric matrix M , it is well known that the corresponding *Laplacian* matrix $L = D - M$ is positive semidefinite [211]. Now we write (2.16) as follows for $\lambda \in \Lambda$:

$$E(S) = \underbrace{S^T(M - \lambda D)S}_{G(S)} + \lambda \underbrace{S^T DS}_{H(S)}. \quad (2.17)$$

H is a unary potential for binary variables: $H(S) = \sum_p d(p)s_p^2 = \sum_p d(p)s_p$. Also, notice that $\forall \lambda \geq 1$, G is concave w.r.t S because $M - \lambda D$ is negative semidefinite (as it is the sum of two negative semidefinite matrices: $M - \lambda D = -L + (1 - \lambda)D$). Therefore, let ∇ denotes the gradient, we have the following pseudo-bounds at current solution S_t which includes bounds of (2.16) for $\lambda \geq 1$.

$$\begin{aligned} \mathcal{F}_t(S, \lambda) &= G(S_t) + \nabla G(S_t)^T(S - S_t) + \lambda H(S) \\ &= \underbrace{G(S_t) - \nabla G(S_t)^T S_t}_{\text{Constant}} + \underbrace{2[(M - \lambda D)S_t]^T S + \lambda H(S)}_{\text{Unary potential}} \end{aligned} \quad (2.18)$$

2.4 Experiments

2.4.1 Appearance entropy based segmentation

Robustness w.r.t initialization and binning: We use GrabCut and BCD interchangeably for the rest of the paper. Left part of Fig. 2.5 depicts an example of interactive segmentation with BCD or our proposed pPBC, and shows that BCD is sensitive to initializations, unlike pPBC. pPBC can even tolerate trivial initialization, see an un-supervised segmentation example in the right of Fig. 2.5.

Furthermore, we observed that with more appearance model variables, namely the number of color bins, BCD is more likely to get stuck in weak local minima. We randomly

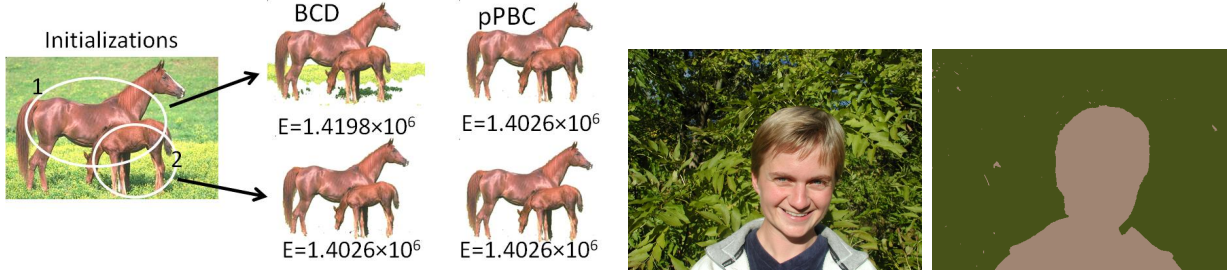


Figure 2.5: Left: interactive segmentations with BCD (GrabCut) or pPBC from different initialization (ellipses). Proposed pPBC method is more robust to inferior initialization. Right: unsupervised figure-ground segmentation with pPBC. Average color is shown.

generated 500 box-like initializations for an input image, and run BCD and pPBC for different numbers of color bins, ranging from 16^3 to 128^3 . From the solutions we obtained with BCD or pPBC, we computed the corresponding error rates and energies. Fig. 2.6 depicts the scatter plots of error rates versus energies for the 500 solutions. Points on bottom-left give low energy and small error rate. The wider these dots spread across the plane, the more local minima the algorithm converged to. pPBC works much better than BCD for finer binning and is more robust to initializations.

Comparisons with the state of the art [220, 234] We compare with GrabCut, which as demonstrated in Sec.2.3.1, can be viewed as a bound optimizer. We run both algorithms on the GrabCut dataset [200] (The cross image excluded for comparison with [234]). We set the weight of the 8-connected contrast-sensitive smoothness term to 15 and vary number of color bins. As shown in Tab.2.3, pPBC consistently gives lower energies and misclassification errors. Our current implementation does not use a straightforward multi-core CPU parallelization of parametric maxflow by breaking the range of λ into intervals.

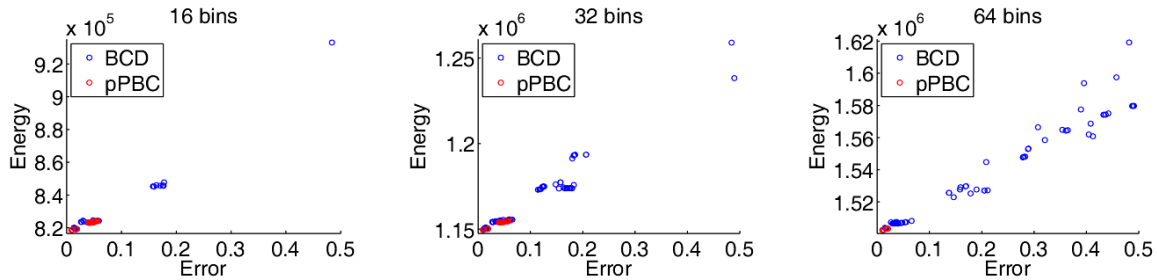


Figure 2.6: Scatter plots; error rates versus energies for 500 solutions of BCD and pPBC.

	Error rate	Time
GrabCut (16 ³ bins)	7.1% ²	1.78 s
GrabCut (32 ³ bins)	8.78%	1.63 s
GrabCut (64 ³ bins)	9.31%	1.64 s
GrabCut (128 ³ bins)	11.34%	1.45 s
DD (16 ³ bins)	10.5%	576 s
One-Cut (16 ³ bins)	8.1%	5.8 s
One-Cut (32 ³ bins)	6.99%	2.4 s
One-Cut (64 ³ bins)	6.67%	1.3 s
One-Cut (128 ³ bins)	6.71%	0.8 s
pPBC (16 ³ bins)	5.80%	11.7 s
pPBC (32 ³ bins)	5.60%	11.9 s
pPBC (64 ³ bins)	5.56%	12.3 s
pPBC (128 ³ bins)	7.51%	15.9 s

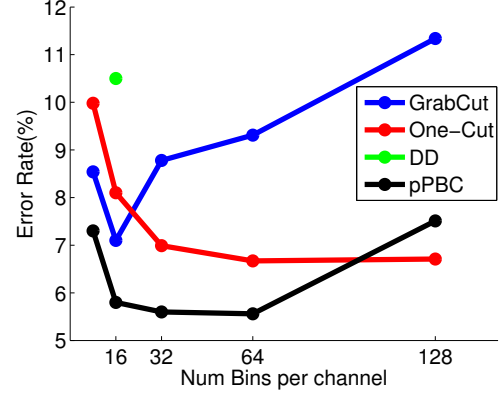


Figure 2.7 & Table 2.2: Error rates and speed on GrabCut dataset for GrabCut [200], Dual Decomposition (DD) [234], One-Cut [220] and our pPBC method.

Thus, significant speed up of our pPBC algorithm is possible. In the next experiment we tuned the smoothness term weight for pPBC and other methods [200, 220] to obtain the best error statistics for each. Fig. 2.7 shows a competitive performance of pPBC.

	Mean Energy	# of lower energy	Mean time(s)
GrabCut [200] - 16 ³ bins	1.2349×10^6	1	1.0s
pPBC - 16 ³ bins	1.2335×10^6	38	11.2s
GrabCut [200] - 32 ³ bins	1.7064×10^6	2	0.9s
pPBC - 32 ³ bins	1.7029×10^6	37	11.7s
GrabCut [200] - 64 ³ bins	2.2408×10^6	1	0.9s
pPBC - 64 ³ bins	2.2361×10^6	47	14.1s

Table 2.3: Statistics of pPBC and GrabCut [200] over the GrabCut database.

Method	KL divergence (2.14)			Bhattacharyya distance (2.12)		
	energy	error	Time	energy	error	Time
Auxiliary Cuts [18]	6189	16.54%	1.8s	-12402	24.1%	1.7s
pPBC($\lambda \leq 0$)	6150	14.88%	N/A	-12451	23.7%	N/A
FTR [91]	5868	7.70%	4.40s	-14499	3.2%	2.71s
pPBC($\lambda \in [-\infty, +\infty]$)	5849	3.63%	2.98s	-14504	2.9%	1.99s

Table 2.4: Matching color distribution (KL or Bhattacharyya distance) with Auxiliary Cuts [18], FTR [91], pPBC and its limited version with $\lambda \leq 0$ for the pseudo-bounds. Mean energy and error are reported.

2.4.2 Matching color distributions

We experiment on the database [200], and used the bounding boxes as initializations. Similar to [18, 91], the target distribution is learned from the ground truth. We compared pPBC with auxiliary cuts [18] and FTR [91]. We also tested a limited version of pPBC where only non-positive λ 's were explored within the family of pseud-bounds. Note that, when λ is non-positive, the parametric family includes only auxiliary functions. The mean error rate, energy and running time are reported in Table 2.4. Exploring only a family of auxiliary functions ($\lambda \leq 0$) did not improve the results. pPBC with parameter $\lambda \in \mathbb{R}$ yielded the best performance, while being slightly slower than auxiliary cuts (even though pPBC explores a family of functions instead of only one). FTR yielded comparable mean energy to pPBC, but is slower. Fig. 2.2 depicts typical examples.

2.4.3 Curvature Regularization

We applied our framework to the curvature model proposed in [72], which penalizes 90 degree angles in a 4-connect neighborhood system. We also compare pPBC to the recent algorithms (LSA-AUX and LSA-TR) in [90], which were shown to outperform standard state-of-the-art methods such as QPBO [198] and TRWS [123]. Fig. 2.8 plots the energies of the solutions with different weights of the curvature term. pPBC-T gives the lowest energy among all methods. We also observed that the best λ for pPBC-T often does not make the pseudo-bound an auxiliary function, which means the bounding constraint is violated.

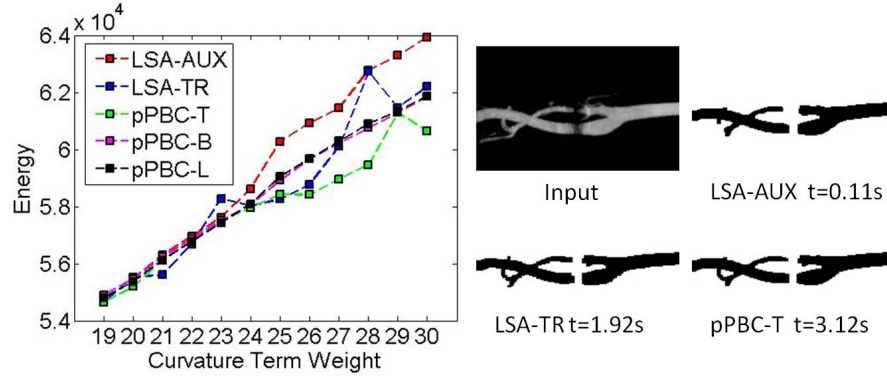


Figure 2.8: Segmentation with the curvature regularization model in [72].

2.4.4 Deconvolution

Fig. 2.9 depicts a binary image convolved with a mean 3×3 filter, with a Gaussian noise added. The purpose is to recover the original image via optimizing the energy: $E(S) = \sum_{p \in \Omega} (I_p - \frac{1}{9} \sum_{q \in \mathcal{N}_p} s_q)^2$, where \mathcal{N}_p is a 3×3 neighborhood window centered at pixel p . In this energy, all pairwise interactions are supermodular. We compared our pPBC-B, L or T to the recent algorithms in [90] (LSA-AUX and LSA-TR). Table 2.5 shows average energy of those methods. Note that LSA-TR achieves lower energy but visually worse deconvolution. For $\sigma = 0.05$ noise, LSA-AUX takes 0.12s, LSA-TR 0.73s and pPBC-T 1.46s.

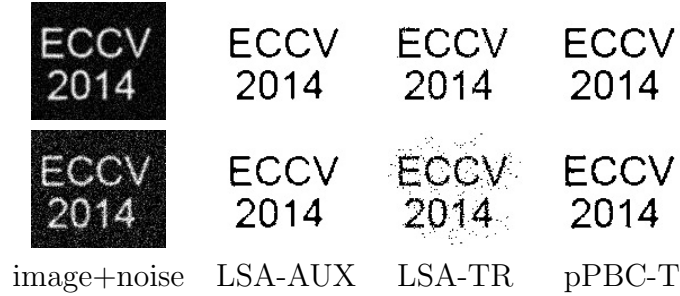


Figure 2.9: Deconvolution results. Top row: noise $\sigma = 0.10$, bottom row: $\sigma = 0.2$.

Noise σ	LSA-AUX [90]	LSA-TR [90]	pPBC-L	pPBC-B	pPBC-T
0.05	40.34	30.83	39.49	39.65	30.81
0.10	130.24	119.84	128.68	128.06	121.20
0.15	277.27	263.06	275.89	276.62	266.35
0.20	482.54	451.78	480.80	482.09	471.11

Table 2.5: Average energy with 10 random noisy images.

2.5 Conclusion

This chapter proposes a new general *pseudo-bound optimization* paradigm for approximate iterative minimization of high-order and non-submodular binary energies. It generalizes the standard *majorize-minimize* principle relaxing the bound constraint for an auxiliary function. We propose to optimize a family of pseudo-bounds at each iteration. To guarantee the energy decreases we include at least one bound in the family. We propose parametric maxflow [125, 105, 81] to explore all global minima for the whole family in low-order polynomial time.

To guarantee polynomial time complexity, pseudo-bounds families with *monotone* dependence on parameter are chosen. We propose and discuss several options of pseudo-bound families for various high-order and non-submodular pairwise energies. Our *parametric Pseudo-Bound Cuts* algorithm (pPBC) improves the-state-of-the-art in many energy minimization problems, e.g. entropy based segmentation, target distributions matching, curvature regularization and deconvolution. In particular, we show that the well-known GrabCut algorithm [200] is a bound optimizer. Our pseudo-bound approach is more robust to inferior initialization and finer binning for image segmentation. Our pPBC algorithm also gives lower energy than *Auxiliary Cuts* [18] and *Fast Trust Region* [91] for distribution matching and other challenging optimization problems in vision.

Chapter 3

Kernel Clustering Meets MRF Regularization

3.1 Introduction: Terminology and Motivation

While independently developed as different methodologies, standard regularization and kernel clustering techniques are based on objective functions with many complementary properties. Our goal is to combine these functions into a joint objective or *energy* applicable to image segmentation or general clustering problems. On the one hand, we show that common regularization methods can use extra terms like *normalized cut* (NC) [211] to enforce balanced partitioning of arbitrary high-dimensional image features, *e.g.* a combination of color, texture, depth, or motion, where *model-fitting* [263, 200] fails, compare Fig. 3.1(b)(e). On the other hand, standard clustering applications can benefit from an inclusion of basic pairwise or higher-order regularization constraints, *e.g.* edge alignment [37, 30], bin-consistency [120], label cost [60]. Regularization and kernel clustering could not be combined before due to optimization difficulties [132].

On a surface, even the formulations of kernel clustering and regularization-based segmentation may seem significantly different. While the general terms *clustering* and *segmentation* are largely synonyms, the latter is more common for images where data points are intensities, colors, or higher dimensional features $I_p \in \mathcal{R}^N$ sampled at regularly placed pixels $p \in \mathcal{R}^M$. For example, the image in Fig. 3.1(a) combines colors and motion vectors into RGBUV features $I_p \in \mathcal{R}^5$ on grid points $p \in \mathcal{R}^2$. The pixels' locations are important. Many *regularization* methods for image segmentation treat I_p as a function $I : \mathcal{R}^M \rightarrow \mathcal{R}^N$ and process domain \mathcal{R}^M (locations) and range \mathcal{R}^N (features) in very different ways. For

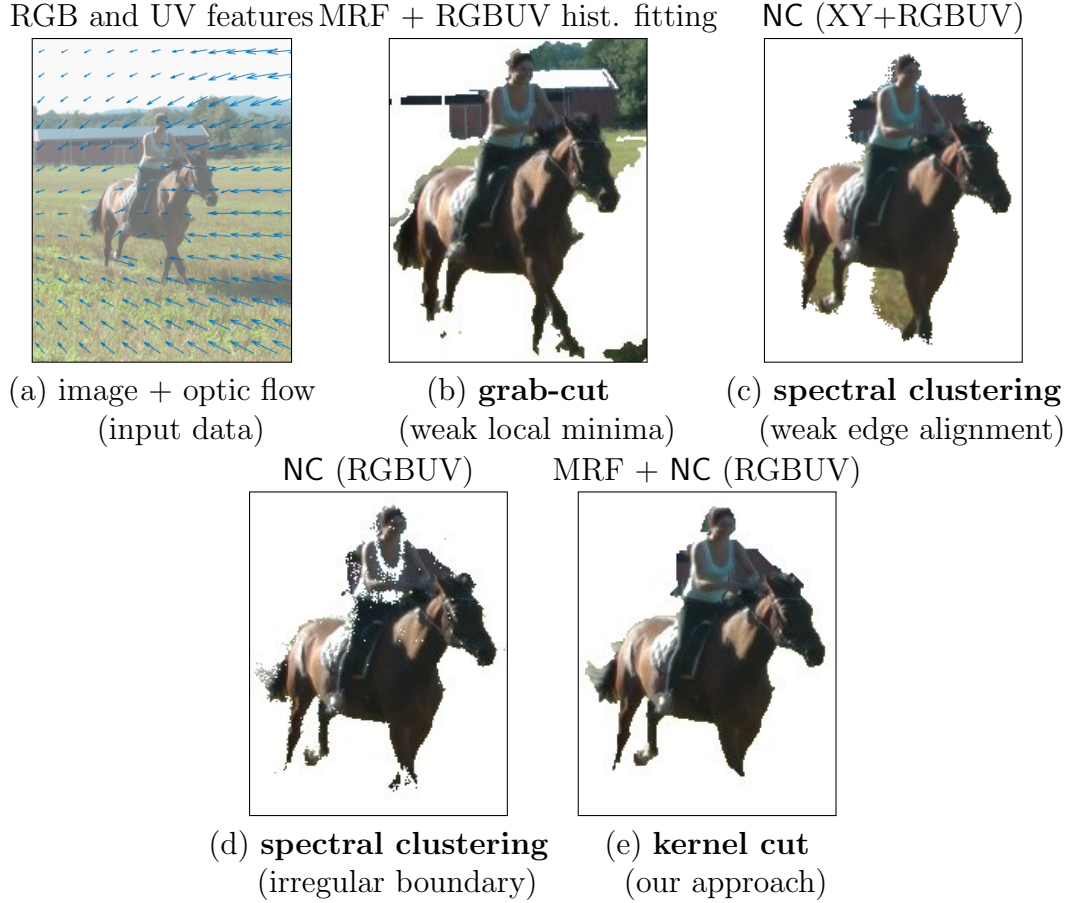


Figure 3.1: Segmentation of 5D image data (a). For higher-dimensional features, regularized model-fitting [263, 60] becomes sensitive to local minima, *e.g.* *grabcut* [200] fitting RGBUV histograms (b). Spectral clustering like *normalized cut* (NC) [211] is scalable to high dimensional features, but it is known for splitting regions (c) or lack of regularity (d). Our *kernel cut* (e) combines *kernel clustering* over arbitrary features with standard regularization in the image domain, see energy (3.1).

example, MRF [82] and variational techniques [166] use pixel locations for geometrically motivated segments' shape priors, while pixel features are used in segments' appearance likelihood models [37, 30, 28, 189], *e.g.* Fig. 3.1(b).

In contrast, *clustering* typically assumes arbitrary data points I_p with non-informative indices p . General clustering techniques [69, 235, 4], *e.g.* *K-means* or spectral methods, apply to images [211, 2] by combining pixel locations with colors or other features into data points I_p in \mathcal{R}^{M+N} . For example, the result in Fig. 3.1(c) uses \mathcal{R}^7 points combining locations XY and RGBUV values. Without the locations the result is spatially noisy (d). We focus on a well-known general group of *kernel clustering* methods [107, 211, 9, 64].

Some differences in formulations of kernel clustering and regularization methods are not essential and easily resolve with proper notation working as a common platform for both (Sec. 3.1.1, Tab. 3.1). Our notation presents spectral clustering as a high-order term in a joint energy making similarities and differences more transparent. Once notation is established, we present our joint energy (3.1) combining kernel clustering and regularization terms, give some specific basic examples (Tab. 3.2), and further motivate our approach. Later background section reviews standard (MRF) regularization and kernel clustering objectives in details and technical sections explain how to optimize the joint energy using new linear bounds for the high-order kernel clustering term.

3.1.1 Notation

Sec. 1.2 has introduced some frequently used notations used in this thesis. Here, we describe extra notations for this Chapter.

Tab. 3.1 summarizes our notations for segmentation. In Tab. 3.1, we give alternative representations of the same notations. Such alternative representation is somewhat superfluous, but it gives flexibility needed for uniting diverse methodologies for segmentation and clustering covered in Sec. 3.2. We equivalently represent segmentation of Ω either as a *labeling* $S := (S_p | p \in \Omega)$ combining integer point labels $1 \leq S_p \leq K$ or as a partitioning $\{S^k\}$ of set Ω into K non-overlapping subsets or segments $S^k := \{p \in \Omega | S_p = k\}$. As a minor abuse of notation, S^k will also be a set indicator vector $\{0, 1\}^{|\Omega|}$. Exact interpretation of S^k is clear from the context. Since our bounds are also useful for relaxation methods, we may discuss *relaxed* segment support vectors S^k in $[0, 1]^{|\Omega|}$.

variable	alternative <i>relaxed</i> representation
S_p	vector $[0, 1]^K$, p -th row of <i>assignment matrix</i> S
S_c	subset of rows of <i>assignment matrix</i> S
S	an <i>assignment matrix</i> $[0, 1]^{ \Omega \times K}$
S_p^k	element of <i>assignment matrix</i> in $[0, 1]$
S^k	vector $[0, 1]^{ \Omega }$, k -th column of <i>assignment matrix</i> S
$S^{k'}$	transposed vector $[0, 1]^{ \Omega }$

Table 3.1: Our notation for segmentation of points $p \in \Omega$ uses discrete labels and binary indicators, see Tab. 1.1. Without much ambiguity, segment S^k could mean both a subset of Ω or its indicator vector, *i.e.* S^k is either an element of *power set* $\mathcal{P}(\Omega)$ or a vector $\{0, 1\}^{|\Omega|}$. While unnecessary for most of the technical results in this paper, in the context of *relaxation* methods it is easy to switch to an alternative representation (the last column) where segment S^k becomes a relaxed vector $[0, 1]^{|\Omega|}$. This is consistent with a common (relaxed) *assignment matrix* representation of segmentation S where integer label S_p becomes a vector on probability simplex Δ^K specifying pixel’s support/distribution over K labels.

3.1.2 Our approach summary

We combine standard kernel (pairwise) clustering criteria such as *Average Association* (AA) or *Normalized Cut* (NC) [211] and common regularization functionals such as MRF potentials [82, 144]. The general form of our joint energy is

$$E(S) = E_A(S) + \gamma \sum_{c \in \mathcal{F}} E_c(S_c) \quad (3.1)$$

where the first term is some kernel clustering objective based on data *affinity matrix* or *kernel* $A := [A_{pq}]$ with elements $A_{pq} := A(I_p, I_q)$ defined by some similarity function $A(\cdot, \cdot)$. The second term in (3.1) is a general formulation of MRF *potentials* [31, 120, 60]. Tab. 3.2 previews basic examples of the terms in joint energy (3.1) using different “graph cut” criteria.

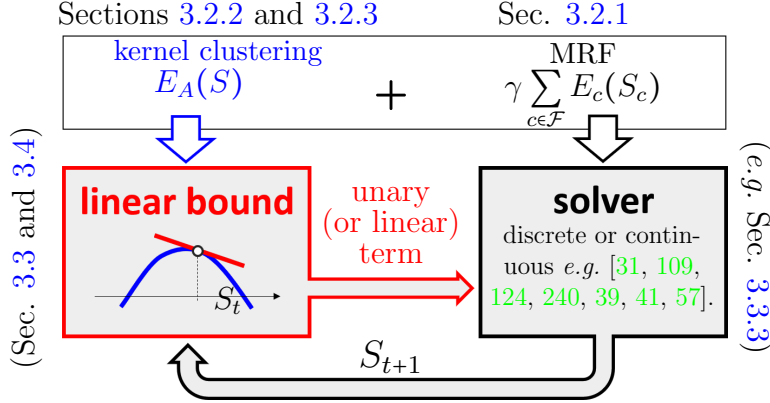


Figure 3.2: Our *Kernel Cut* approach to minimizing energy (3.1). Standard (MRF) regularization solvers can easily integrate our linear *kernel* or *spectral* bounds for the clustering term (Sec.3.3,3.4) producing an iterative *bound optimization* for (3.1).

Constant γ in (3.1) is a relative weight of the (MRF) regularization term. Subset $c \subseteq \Omega$ represents a *factor* often consisting of nearby pixels. Factor labels $S_c := (S_p | p \in c)$ is a *restriction* of labeling S to c . Potentials $E_c(S_c)$ for a given set of factors \mathcal{F} represent various unary, second, or higher order constraints, where factor size $|c|$ defines the order. The left column in Tab. 3.2 is an example of the second-order *Potts model* that can be equivalently written as a quadratic function. Factor features $\{I_p | p \in c\}$ often work as parameters for potentials E_c . For example, $w_{pq} = w(I_p, I_q)$ is a common way to set pairwise penalties in Tab. 3.2 (left column).

Typical kernel clustering methods encourage balanced segments using ratio-based objectives E_A as in Tab. 3.2 (right column). Due to normalization, such objectives can be seen as high-order potentials of order $|\Omega|$ that are difficult to optimize. Sections 3.2.2, 3.2.3 review popular kernel clustering criteria and standard approximate optimization methods.

In order to optimize the combination of kernel clustering term E_A with regularization constraints in energy (3.1), we propose two unary (linear) bounds for E_A . Such bounds are easy to integrate into many existing regularization solvers as outlined in Fig. 3.2. In general, the second term in (3.1) could be any discrete or continuous objective with a good solver. We focus on discrete (MRF/CRF) regularization potentials in (3.1) only to be specific and because the code for the corresponding solvers is widely available. The following two subsections summarize the motivation and the main technical contributions of this paper.

regularization terms $\sum_{c \in \mathcal{F}} E_c(S_c)$	kernel clustering terms $E_A(S)$
$\sum_{pq} w_{pq} [S_p \neq S_q] \equiv \frac{1}{2} \sum_k S^{k'} \mathcal{W}(\mathbf{1} - S^k)$ <p style="text-align: center;"><i>multi-way cut</i> a.k.a. Potts model [31], see (3.2)</p>	$\sum_k \frac{S^{k'} A(\mathbf{1} - S^k)}{ S^k } \text{ or } \sum_k \frac{S^{k'} A(\mathbf{1} - S^k)}{d' S^k}$ <p style="text-align: center;"><i>average and normalized cuts</i> [211] see (3.37) and (3.38)</p>

Table 3.2: Examples of “**graph cut**” criteria appearing in the contexts of (MRF) *regularization* and *kernel clustering* that can be used in joint energy (3.1) simultaneously. The cut cost, *i.e.* the sum of edge weights w_{pq} or affinities A_{pq} between the segments, can be represented via matrices $\mathcal{W} = [w_{pq}]$ or $A = [A_{pq}]$, and segment indicators S^k , see Tab. 3.1. The right column differs only by normalization over segment *cardinality* $|S^k|$ or *weighted cardinality* $d' S^k$ where $d := A\mathbf{1}$ are *node degrees*.

3.1.3 Motivation and Related work

Due to significant differences in their existing optimization methods, kernel clustering (*e.g.* NC) and regularization methods (*e.g.* MRF) are used separately in unsupervised or weakly-supervised applications of vision and learning. They have complementary strengths and weaknesses.

For example, NC optimizes a balanced kernel clustering criterion based on a kernel (affinities) between any high-dimensional features [211, 155, 6]. In contrast, regularization methods for unsupervised or weakly-supervised image segmentation typically combine constraints on segments shapes with *probabilistic K-means* [114] or explicit *model fitting* over segments features [42, 263, 200, 60]. Fitting parametric models seems viable when data in each segment supports a simple model, *e.g.* Gaussian [42] or line/plane [60]. But, if segment’s data is arbitrarily complex, the corresponding model should be sufficiently general in order to represent such complexities. Thus, image segmentation of generic objects requires fitting models like histograms or GMMs [263, 200]. This results in over-fitting, see Fig. 3.1(b). Indeed, we show that such over-fitting happens even for low dimensional color features [220], see Fig. 3.3(b,e) and Fig. 3.4(b). Our joint energy (3.1) allows to combine regularization of segments shapes with unsupervised kernel-based clustering of arbitrarily complex segments features. In general, kernel-based clustering methods are a prevalent choice in the learning community as model fitting (*e.g.* EM) becomes intractable in high dimensions. Sec. 3.5.2 shows potent segmentation results for basic examples of energy (3.1)

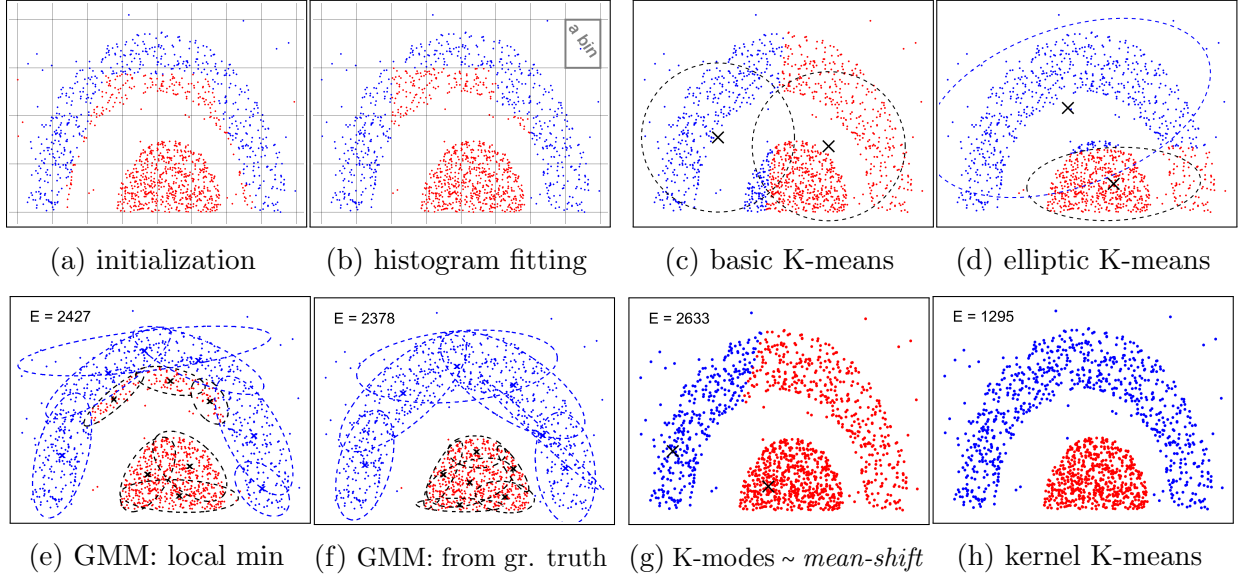


Figure 3.3: *Model fitting (pKM) (3.13) vs kernel K-means (kKM) (3.21)*. Histogram fitting converges in one step assigning initially dominant bin label (a) to all points in the bin (b): energy (3.13,3.14) is minimal at any volume-balanced solution with one label inside each bin [114]. Basic and elliptic K-means (one mode GMM) under-fit the data (c,d). Six mode GMMs over-fit (e) as in (b). GMMs have local minima issues; ground-truth initialization (f) yields lower energy (3.13,3.14). Kernel K-means (3.20,3.21) with Gaussian kernel k in (h) outperforms pKM with distortion $\|\cdot\|_k$ in (g) related to K-modes or mean-shift (*weak kKM*, see Sec.3.2.2).

with features like RGBXY (color + location), RGBD (color + depth), RGBUV (color + motion) where regularized model-fitting methods fail.

Standard applications of kernel clustering methods can also benefit from regularization constraints [252, 73, 48]. For example, NC approach to image segmentation is known for weak alignment to contrast boundaries [6], see Fig. 3.1(cd). Adding the standard contrast-sensitive Potts (regularization) term [31, 30] offers a principled solution, see Fig. 3.1(e). We also show benefits from combining NC with higher-order constraints, such as sparsity or label costs [60]. For example, P^n -Potts regularization [120] can enforce tag-consistency in the context of image database clustering. Sec. 3.5.1 shows many proof-of-the-concept examples.

Kernel clustering vs. Potts model: Kernel clustering objectives E_A (Sections 3.2.2, 3.2.3) in our joint energy (3.1) can be juxtaposed with the most basic MRF regularizer,

the Potts model (Sec. 3.2.1), *e.g.* compare two columns in Tab. 3.2. Kernel clustering and Potts regularization minimize the sum of weighted edges between segments on a given graph. Both corresponding objectives are often called “pairwise” or “cuts”. The main difference is that clustering criteria E_A normalize the sum of edge weights to encourage balanced partitioning, while the Potts model minimizes the sum “as is” to reduce segmentation boundary length. Due to normalization, E_A is a hard-to-optimize high-order term in energy (3.1).

Both kernel clustering and Potts model objectives are defined by the graph connectivity and/or the corresponding edge weights or affinities, *e.g.* w_{pq} or A_{pq} in Sections 3.2.1 and 3.2.2. It is usual to set the neighborhood and edge weights based on specific features, criteria, and application. For instance, Potts model over nearest-neighbor pixel grid defines first-order geometric shape priors [28], while an example of larger connectivity Potts is *dense CRF* [130, 231]. All Potts models lack balancing. Their minimization results in a trivial solution unless there are some additional constraints, *e.g.* volumetric or data likelihood terms (Sec.3.2.1).

Kernel clustering criteria E_A normally use dense graphs. But unlike dense CRF or any other Potts model, the corresponding ratio-based objectives are designed for *unsupervised* balanced partitioning that does not require any known or estimated data likelihood models.

3.1.4 Main contributions

Our energy (3.1) combines standard concepts in unsupervised learning with regularization methodologies common in computer vision. Previous efforts [132] combining kernel clustering (*e.g.* NC) with the Potts model significantly altered the latter to make it fit the standard trace-based formulation of NC, see Sec.3.2.4. In contrast, we propose a general *majorize-minimize* optimization principle directly integrating our new unary/linear bounds for kernel clustering objectives E_A into existing powerful solvers for Potts or other regularization models. Examples of such solvers are combinatorial [31, 109], LP relaxation [124, 240], mean field approximation [130], or TV-based [39, 41, 57] methods.

Our preliminary results appear in [218] and [222]. The main contributions of our work are summarized below:

- We propose a general multi-label segmentation or clustering energy (3.1) combining kernel clustering (*e.g.* NC) with second or higher-order regularization (*e.g.* MRF). The clustering term can enforce balanced partitioning of observed features and MRF or other terms can enforce regularization constraints. In particular, including balanced

kernel clustering term is a robust well-motivated alternative to model-fitting terms [263, 200], which fail on higher dimensional image features.

- We use a *concave relaxation* to derive two types of unary (linear) upper bounds for several classes of kernel clustering criteria E_A . The two types are *kernel bound* (exact) and *spectral bound*¹ (approximate). Interestingly, optimizing our linear bounds for $E_A(S)$ (no other terms) over discrete segmentation variables $S^k \in \{0, 1\}^{|\Omega|}$ is equivalent to iterative *kernel K-means* or *K-means* discretization heuristic in spectral relaxation methods.
- Our unary/linear bounds for E_A give solvable *auxiliary functions* for joint energy (3.1) as long as its second term has a solver that can integrate extra unary/linear potentials, see Fig. 3.2. For example, the second term can be any regularization potentials solvable by discrete (*e.g.* message passing, relaxations, mean-field approximations) or continuous (*e.g.* convex, primal-dual) algorithms. In the context of standard pairwise and higher-order MRF potentials we demonstrate move-making algorithms generalizing α -expansion and $\alpha\beta$ -swap moves to energy (3.1).
- As our experiments show, typical applications of kernel clustering (*e.g.* NC) can benefit from extra MRF constraints. MRF segmentation also benefits from kernel clustering terms encouraging balanced partitioning of object features. In particular, NC+MRF framework scales to object segmentation with higher-dimensional image features (*e.g.* RGBXY, RGBD, RGBM) where standard regularization methods with model-fitting [263, 200, 60] fail.

The rest of the paper is organized as follows. Background Section 3.2 starts from reviewing standard (MRF) regularization models for segmentation. Due to importance for our work, Sec. 3.2 also covers the basics of clustering from *K-means* to its powerful kernel-based generalizations, including normalized cut (NC). The main technical Sections 3.3 and 3.4 present our kernel and spectral bounds for standard kernel clustering objectives E_A . They also discuss combinatorial move making graph cut algorithms using such unary/linear bounds for optimizing joint energy (3.1) combining E_A with MRF regularization constraints. Sec. 3.5 presents many experiments where either standard kernel clustering methods benefit from additional MRF constraints or common applications of MRF benefit from an additional kernel clustering term for various high-dimensional image features.

¹Here *spectral bound* means *spectral auxiliary function* in the context of optimization, not to be confused with bounds on eigenvalues.

3.2 Background on Regularization and Clustering

3.2.1 Overview of MRF regularization

Probably the most basic MRF regularization potential corresponds to the pairwise (second-order) Potts model [31] used for segmentation boundary smoothness and **edge alignment**

$$\sum_{c \in \mathcal{F}} E_c(S_c) = \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q] \approx \|\partial S\| \quad (3.2)$$

where a set of pairwise factors $\mathcal{F} = \mathcal{N}$ includes *edges* $c = \{pq\}$ between pairs of neighboring nodes and $[\cdot]$ are *Iverson brackets*. Weight w_{pq} is a discontinuity penalty between p and q . It could be a constant or may be set by a decreasing function of intensity difference $I_p - I_q$ attracting the segmentation boundary to image contrast edges in (1.7) [30]. This is similar to the image-based boundary length in geodesic contours [37, 28].

A useful **bin consistency** constraint enforced by the P^n -Potts model [120] is defined over an arbitrary collection of high-order factors \mathcal{F} . Factors $c \in \mathcal{F}$ correspond to predefined subsets of nodes such as *superpixels* [120] or *bins* of pixels with the same color/feature [183, 220]. The model penalizes inconsistency in segmentation of each factor

$$\sum_{c \in \mathcal{F}} E_c(S_c) = \sum_{c \in \mathcal{F}} \min\{T, |c| - |S_c|^*\} \quad (3.3)$$

where T is some threshold and $|S_c|^* := \max_k |S^k \cap c|$ is the cardinality of the largest segment inside c . Potential (3.3) has its lowest value (zero) when all nodes in each factor are within the same segment.

Standard **label cost** [60] is a sparsity potential defined for a single high-order factor $c = \Omega$. In its simplest form it penalizes the number of distinct segments (labels) in S

$$E_\Omega(S) = \sum_k h_k \cdot [|S^k| > 0] \quad (3.4)$$

where h_k could be a constant or a cost for each specific label.

Potentials (3.2), (3.3), (3.4) are only a few examples of regularization terms widely used in combination with powerful discrete solvers like graph cut [31], belief propagation [248], TRWS [124], LP relaxation [240, 113], or continuous methods [39, 41, 57].

Image segmentation methods often combine regularization with a **likelihood term** integrating segments/objects color models. For example, [30, 29] used graph cuts to combine second-order edge alignment (3.2) with a unary (first-order) appearance term

$$- \sum_k \sum_{p \in S^k} \log P^k(I_p) \quad (3.5)$$

where $\{P^k\}$ are given probability distributions. Unary terms like (3.5) are easy to integrate into any of the solvers above. If unknown, parameters of the models $\{P^k\}$ in a regularization energy including (3.5) are often estimated by iteratively minimizing the energy with respect to S and model parameters [263, 42, 8, 200, 60]. In presence of variable model parameters, (3.5) can be seen as a *maximum likelihood* (ML) model-fitting term or a *probabilistic K-means* clustering objective [114]. The next section reviews K-means and other standard clustering methods.

3.2.2 Overview of K-means and clustering

Many clustering methods are based on K-means (KM). The most basic iterative KM algorithm [69] can be described as the *block-coordinate descent* for the *mixed* objective (1.1)

$$F_{km}(S, \mathbf{m}) := \sum_k \sum_{p \in S^k} \|I_p - m_k\|^2 \quad (3.6)$$

combining discrete variables $S = \{S^k\}_{k=1}^K$ with continuous variables $\mathbf{m} = \{m_k\}_{k=1}^K$ representing cluster “centers”. For any given S the optimal centers $\arg \min_{\mathbf{m}} F_{km}(S, \mathbf{m})$ are the means

$$\mu_{S^k} := \frac{\sum_{q \in S^k} I_q}{|S^k|} \quad (3.7)$$

where $|S^k|$ is the segment’s cardinality. The greedy KM procedure (1.18) converges only to a local minimum of KM objective (3.6), which is known to be NP hard to optimize. There are also other approximation methods. Below we review the properties of KM objective (3.6) independently of optimization.

The optimal centers m_k in (3.7) allow to represent (3.6) via an equivalent objective of a single argument S

$$\sum_k \sum_{p \in S^k} \|I_p - \mu_{S^k}\|^2 \equiv \sum_k |S^k| \cdot \text{var}(S^k). \quad (3.8)$$

The sum of squared distances between data points $\{I_p | p \in S^k\}$ and mean μ_{S^k} normalized by $|S^k|$ gives the *sample variance* denoted by $\text{var}(S^k)$. Formulation (3.8) presents the basic KM objective as a standard *variance criterion* for clustering. That is, K-means attempts to find K compact clusters with small variance.

K-means can also be presented as a “pairwise” or kernel clustering criteria with Euclidean affinities. The *sample variance* can be expressed as the sum of distances between all pairs of the points. For example, plugging (3.7) into (3.8) reduces this KM objective to

$$\sum_k \frac{\sum_{p,q \in S^k} \|I_p - I_q\|^2}{2 |S^k|}. \quad (3.9)$$

Taking the square in the nominator transforms (3.9) to another equivalent KM energy with Euclidean dot-product affinities

$$\stackrel{c}{=} - \sum_k \frac{\sum_{pq \in S^k} \langle I_p, I_q \rangle}{|S^k|}. \quad (3.10)$$

Note that we use $\stackrel{c}{=}$ and $\stackrel{c}{\approx}$ for “up to additive constant” relations.

Alternatively, K-means clustering can be seen as Gaussian model fitting. Formula (3.5) for normal distributions with variable means m_k and some fixed variance

$$- \sum_k \sum_{p \in S^k} \log N(I_p | m_k) \quad (3.11)$$

equals objective (3.6) up to a constant.

Various extensions of objectives (3.6), (3.8), (3.9), (3.10), or (3.11) lead to many powerful clustering methods such as kernel K-means, average association, and Normalized Cut, see Tab. 3.3. In the following, we discuss extensions of the basic K-means to probabilistic K-means and kernel K-means, see a summary in Tab. 3.3.

Probabilistic K-means (pKM) and model fitting

One way to generalize K-means is to replace squared Euclidean distance in (3.6) by other *distortion* measures $\|\cdot\|_d$ leading to a general *distortion energy* commonly used for clustering

$$\sum_k \sum_{p \in S^k} \|I_p - m_k\|_d. \quad (3.12)$$

The optimal value of parameter m_k may no longer correspond to a *mean*. For example, the optimal m_k for non-squared L_2 metric is a *geometric median*. For exponential distortions the optimal m_k may correspond to *modes* [202, 36], see [221, Appendix B].

A seemingly different way to generalize K-means is to treat both means and covariance matrices for the normal distributions in (3.11) as variables. This corresponds to the standard *elliptic K-means* [215, 201, 60]. In this case variable model parameters $\theta_k = \{m_k, \Sigma_k\}$ and data points I_p are not in the same space. Yet, it is still possible to present elliptic K-means as distortion clustering (3.12) with “distortion” between I_p and θ_k defined by an operator $\|\cdot\|_d$ corresponding to a likelihood function

$$\|I_p - \theta_k\|_d := -\log N(I_p | \theta_k).$$



A. basic K-means (KM) (e.g. [69])	
$ \begin{aligned} & \sum_k \sum_{p \in S^k} \ I_p - \mu_{S^k}\ ^2 \\ &= \sum_k \frac{\sum_{pq \in S^k} \ I_p - I_q\ ^2}{2 S^k } \\ &\stackrel{c}{=} - \sum_k \frac{\sum_{pq \in S^k} \langle I_p, I_q \rangle}{ S^k } \\ &\stackrel{c}{=} - \sum_k \sum_{p \in S^k} \ln \mathcal{N}(I_p \mu_{S^k}) \end{aligned} $	Variance criterion $\sum_k S^k \cdot \text{var}(S^k)$
 more complex probability models	more complex data representation 
B. probabilistic K-means (pKM)	C. kernel K-means (kKM)
<i>equivalent energy formulations:</i> $\sum_k \sum_{p \in S^k} \ I_p - \theta_k\ _d = - \sum_k \sum_{p \in S^k} \ln \mathcal{P}(I_p \theta_k)$	<i>equivalent energy formulations:</i> $ \begin{aligned} \sum_k \sum_{p \in S^k} \ \phi(I_p) - \mu_{S^k}\ ^2 &= \sum_k \frac{\sum_{pq \in S^k} \ I_p - I_q\ _k^2}{2 S^k } \\ &\stackrel{c}{=} - \sum_k \frac{\sum_{pq \in S^k} k(I_p, I_q)}{ S^k } \end{aligned} $
<i>related examples:</i> elliptic K-means [215, 201] geometric model fitting [60] K-modes [202] or mean-shift [53] Entropy criterion $\sum_k S^k \cdot H(S^k)$ [263] for highly descriptive models (GMMs, histograms)	<i>related examples:</i> Average Association or Distortion [197] Average Cut [211] Normalized Cut [211, 64] Gini criterion $\sum_k S^k \cdot G(S^k)$ [32] for small-width normalized kernels [158]

Table 3.3: *K-means and related clustering criteria*: Basic K-means (A) minimizes clusters variances. It works as Gaussian model fitting. Fitting more complex models like elliptic Gaussians [215, 201, 60], exponential distributions [8], GMM or histograms [263, 200] corresponds to *probabilistic K-means* [114] in (B). Kernel clustering via *kernel K-means* (C) using more complex data representation.

Similar distortion measures can be defined for arbitrary probability distributions with any variable parameters θ_k . Then, distortion clustering (3.12) generalizes to ML model fitting objective

$$\sum_k \sum_{p \in S^k} \|I_p - \theta_k\|_d \equiv - \sum_k \sum_{p \in S^k} \log P(I_p | \theta_k) \quad (3.13)$$

which is (3.5) with explicit model parameters θ_k . This formulation suggests *probabilistic K-means*² (pKM) as a good idiomatic name for ML model fitting or distortion clustering (3.12), even though the corresponding parameters θ_k are not “means”, in general.

Probabilistic K-means (3.13) is used in image segmentation with models such as elliptic Gaussians [215, 201, 60], gamma/exponential [8], or other generative models [163]. Zhu-Yuille [263] and GrabCut [200] use pKM with highly descriptive probability models such as GMM or histograms. Information theoretic analysis in [114] shows that in this case pKM objective (3.13) reduces to the standard *entropy criterion* for clustering

$$\sum_k |S^k| \cdot H(S^k) \quad (3.14)$$

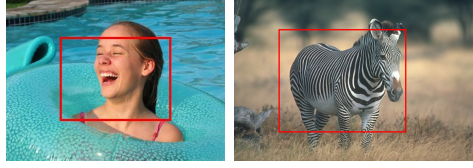
where $H(S^k)$ is the distribution entropy for $\{I_p | p \in S^k\}$.

Intuitively, minimization of the entropy criterion (3.14) favors clusters with tight or “peaked” distributions. This criterion is widely used in categorical clustering [145] and decision trees [32, 151] where the entropy evaluates histograms over “naturally” discrete features. However, the entropy criterion with either discrete histograms or continuous GMM densities has limitations in the context of *continuous* feature spaces, see [221, Appendix C]. Iterative fitting of descriptive models is highly sensitive to local minima [220, 217] and easily over-fits even low dimensional features in \mathcal{R}^2 (Fig. 3.3b,e) or in \mathcal{R}^3 (RGB colors, Fig. 3.4b). This may explain why this approach to clustering is not too common in the learning community. As proposed in (3.1), instead of entropy criterion we will combine MRF regularization with general kernel clustering objectives E_A widely used for balanced partitioning of arbitrary high-dimensional features [211].

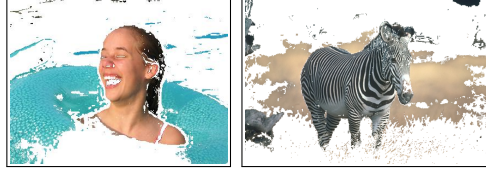
Kernel K-means and related “pairwise” clustering

This section reviews pairwise extensions of K-means (3.10) such as *kernel K-means* (kKM) and related kernel clustering criteria. In machine learning, kKM is a well established data

²The name *probabilistic K-means* in the general clustering context was coined by [114]. They formulated (3.13) after representing distortion energy (3.12) as ML fitting of Gibbs models $\frac{1}{Z_d} e^{-\|x-m\|_d}$ for arbitrary integrable metrics.



(a) Input and initialization



(b) GMM fitting in RGB (GrabCut without edges)



(c) Normalized Cut in RGB

Figure 3.4: Without edge alignment (3.2) GMM-fitting [200] shows stronger data over-fitting compared to kernel clustering [211].

clustering technique [228, 165, 83, 64, 49, 110] that can identify non-linearly separable structures. In contrast to **pKM** based on complex models, k **KM** corresponds to complex (nonlinear) mappings

$$\phi : \mathcal{R}^N \rightarrow \mathcal{H}$$

embedding data $\{I_p | p \in \Omega\} \subset \mathcal{R}^N$ as points $\phi_p \equiv \phi(I_p)$ in a higher-dimensional Hilbert space \mathcal{H} . The original non-linear problem can often be solved by simple linear separators of the embedded points $\{\phi_p | p \in \Omega\} \subset \mathcal{H}$. Kernel K-means corresponds to the basic K-means (3.6) in the embedding space

$$F_{km}(S, \mathbf{m}) = \sum_k \sum_{p \in S^k} \|\phi_p - m_k\|^2. \quad (3.15)$$

Optimal segment centers m_k corresponding to the means

$$\mu_{S^k} = \frac{\sum_{q \in S^k} \phi_q}{|S^k|}. \quad (3.16)$$

reduce (3.15) to k KM energy of the single variable S similar to (3.8)

$$F(S) = \sum_k \sum_{p \in S^k} \|\phi_p - \mu_{S^k}\|^2. \quad (3.17)$$

Similarly to (3.9) and (3.10) one can write kernel clustering criteria equivalent to (3.17) based on Euclidean distances $\|\phi(I_p) - \phi(I_q)\|$ or inner products $\langle \phi(I_p), \phi(I_q) \rangle$, which are commonly represented via *kernel function* $k(x, y)$

$$k(x, y) := \langle \phi(x), \phi(y) \rangle. \quad (3.18)$$

The (non-linear) kernel function $k(x, y)$ corresponds to the inner product in \mathcal{H} . It also defines *Hilbertian metric*³

$$\begin{aligned} \|x - y\|_k^2 &:= \|\phi(x) - \phi(y)\|^2 \\ &\equiv k(x, x) + k(y, y) - 2k(x, y) \end{aligned} \quad (3.19)$$

isometric to the Euclidean metric in the embedding space. Then, pairwise formulations (3.9) and (3.10) for K-means in the embedding space (3.17) can be written for the original data points using isometric kernel distance $\|\cdot\|_k^2$ in (3.19)

$$F(S) \equiv \sum_k \frac{\sum_{pq \in S^k} \|I_p - I_q\|_k^2}{2|S^k|} \quad (3.20)$$

or using kernel function k in (3.18)

$$F(S) \stackrel{c}{=} - \sum_k \frac{\sum_{pq \in S^k} k(I_p, I_q)}{|S^k|}. \quad (3.21)$$

The definition of kernel k in (3.18) requires embedding ϕ . Since pairwise objectives (3.20) and (3.21) are defined for any kernel function in the original data space, it is possible to formulate k KM by directly specifying an affinity function or kernel $k(x, y)$ rather than embedding $\phi(x)$. This is typical for k KM explaining why the method is called *kernel K-means* rather than *embedding K-means*⁴.

Given embedding ϕ , kernel function k defined by (3.18) is *positive semi-definite* (p.s.d), that is $k(x, y) \geq 0$ for any x, y . Moreover, *Mercer's theorem* [165] states that p.s.d. condition

³These can be isometrically embedded into a Hilbert space [103].

⁴This could be a name for some clustering techniques constructing explicit embeddings [14, 253] instead of working with pairwise affinities/kernels.

for any given kernel $k(x, y)$ is sufficient to guarantee that $k(x, y)$ is an inner product in some Hilbert space. That is, it guarantees existence of some embedding $\phi(x)$ such that (3.18) is satisfied. Therefore, k KM objectives (3.17), (3.20), (3.21) are equivalently defined either by embeddings ϕ or p.s.d. kernels k . Thus, kernels are commonly assumed p.s.d. However, as discussed later, kernel clustering objective (3.21) is also used with non p.s.d. affinities.

To optimize k KM objectives (3.17), (3.20), (3.21) one can use the basic KM procedure (1.18) iteratively minimizing mixed objective (3.15) explicitly using embedding ϕ

$$\left(\begin{array}{c} \text{explicit} \\ \text{kKM} \\ \text{procedure} \end{array} \right) \quad S_p \leftarrow \arg \min_k \|\phi_p - \mu_{S_t^k}\| \quad (3.22)$$

where $\mu_{S_t^k}$ is the mean (3.16) for current segment S_t^k . Equivalently, this procedure can use kernel k instead of ϕ . Indeed, as in Section 8.2.2 of [210], the square of the objective in (3.22) is

$$\|\phi_p\|^2 - 2\phi_p' \mu_{S_t^k} + \|\mu_{S_t^k}\|^2 = -2 \frac{\phi_p' \phi S_t^k}{|S_t^k|} + \frac{S_t^{k'} \phi' \phi S_t^k}{|S_t^k|^2}$$

where we use segment S^k as an indicator vector, embedding ϕ as an *embedding matrix* $\phi := [\phi_p]$ where points $\phi_p \equiv \phi(I_p)$ are columns, and $'$ denotes the transpose. Since the crossed term is a constant at p , the right hand side gives an equivalent objective for computing S_p in (3.22). Using *kernel matrix* $\mathcal{K} := \phi' \phi$ and indicator vector $\mathbf{1}_p$ for element p we get

$$\left(\begin{array}{c} \text{implicit} \\ \text{kKM} \\ \text{procedure} \end{array} \right) \quad S_p \leftarrow \arg \min_k \frac{S_t^{k'} \mathcal{K} S_t^k}{|S_t^k|^2} - 2 \frac{\mathbf{1}_p' \mathcal{K} S_t^k}{|S_t^k|} \quad (3.23)$$

where the kernel matrix is directly determined by kernel k

$$\mathcal{K}_{pq} \equiv \phi_p' \phi_q = \langle \phi_p, \phi_q \rangle = k(I_p, I_q).$$

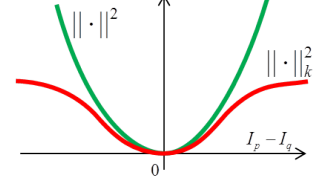
Approach (3.23) has quadratic complexity $\mathcal{O}(|\Omega|^2)$ iterations. But, it avoids explicit high-dimensional embeddings ϕ_p in (3.22) replacing them by kernel k in all computations, *a.k.a.* the *kernel trick*.

Note that the implicit k KM procedure (3.23) is guaranteed to decrease pairwise k KM objectives (3.20) or (3.21) only for p.s.d. kernels. Indeed, equation (3.23) is derived from the standard greedy K-means procedure in the embedding space (3.22) assuming kernel (3.18). The backward reduction of (3.23) to (3.22) can be done only for p.s.d. kernels k when Mercer's theorem guarantees existence of some embedding ϕ such that $k(I_p, I_q) = \langle \phi(I_p), \phi(I_q) \rangle$.

Pairwise energy (3.20) helps to explain the positive result for k KM with common Gaussian kernel $k = \exp \frac{-(I_p - I_q)^2}{2\sigma^2}$ in Fig. 3.3(h). Gaussian kernel distance (red plot below)

$$\|I_p - I_q\|_k^2 \propto 1 - k(I_p, I_q) = 1 - \exp \frac{-(I_p - I_q)^2}{2\sigma^2} \quad (3.24)$$

is a “robust” version of Euclidean metric (green) in basic K-means (3.9). Thus, Gaussian k KM finds clusters with small local variances, Fig. 3.3(h). In contrast, basic K-means (c) tries to find good clusters with small global variances, which is impossible for non-compact clusters.



Average association (AA) or distortion (AD): Equivalent pairwise objectives (3.20) and (3.21) suggest natural extensions of k KM. For example, one can replace Hilbertian metric $\|\cdot\|_k^2$ in (3.20) by an arbitrary zero-diagonal distortion matrix $D = [D_{pq}]$ generating *average distortion* (AD) energy

$$E_{ad}(S) := \sum_k \frac{\sum_{pq \in S^k} D_{pq}}{2|S^k|} \quad (3.25)$$

reducing to k KM energy (3.20) for $D_{pq} = \|I_p - I_q\|_k^2$. Similarly, p.s.d. kernel k in (3.21) can be replaced by an arbitrary pairwise similarity or affinity matrix $A = [A_{pq}]$ defining standard *average association* (AA) energy

$$E_{aa}(S) := - \sum_k \frac{\sum_{pq \in S^k} A_{pq}}{|S^k|} \quad (3.26)$$

reducing to k KM objective (3.21) for $A_{pq} = k(I_p, I_q)$. We will also use association between any two segments S^i and S^j

$$assoc(S^i, S^j) := \sum_{p \in S^i, q \in S^j} A_{pq} \equiv S^{i'} A S^j \quad (3.27)$$

allowing to rewrite AA energy (3.26) as

$$E_{aa}(S) \equiv - \sum_k \frac{assoc(S^k, S^k)}{|S^k|} \equiv - \sum_k \frac{S^{k'} A S^k}{\mathbf{1}' S^k} \quad (3.28)$$

The matrix expressions in (3.27) and (3.28) represent segments S^k as indicator vectors such that $S_p^k = 1$ iff $S_p = k$ and symbol $'$ means a transpose. Matrix notation as in (3.28) will be used for all kernel clustering objectives in this paper.

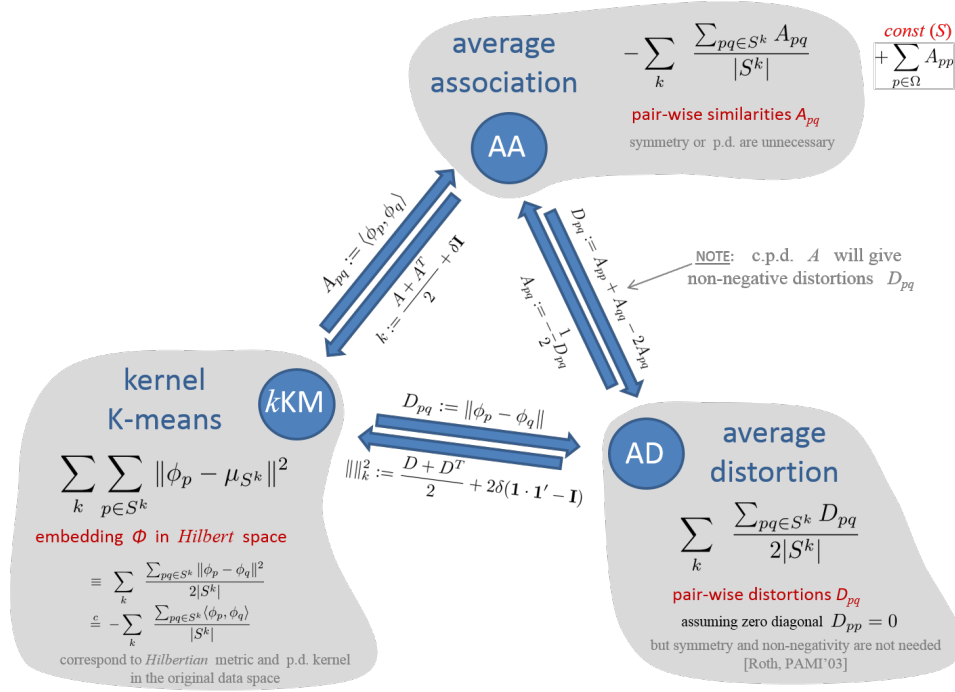


Figure 3.5: Equivalence of kernel clustering methods: *kernel K-means* ($k\text{KM}$), *average distortion* (AD), *average association* (AA) based on Roth et al. [197], see (3.29), (3.30). Equivalence of these methods in the general *weighted* case is discussed in [221, Appendix A, Fig.33].

$k\text{KM}$ algorithm (3.23) is not guaranteed to decrease (3.26) for improper (non p.s.d.) kernel matrix $\mathcal{K} = A$, but general AA and AD energies could be useful despite optimization issues. However, [197] showed that dropping metric and proper kernel assumptions are not essential; there exist p.s.d. kernels with $k\text{KM}$ energies equivalent (up to constant) to AD (3.25) and AA (3.26) for arbitrary associations A and zero-diagonal distortions D , see Fig. 3.5.

For example, for any affinity A in (3.26) the *diagonal shift* trick of Roth et al. [197] generates the “kernel matrix”

$$\mathcal{K} = \frac{A + A'}{2} + \delta \cdot \mathbf{I}. \quad (3.29)$$

For sufficiently large scalar δ matrix \mathcal{K} is positive definite yielding a proper discrete kernel $k(I_p, I_q) \equiv \mathcal{K}_{pq}$

$$k(I_p, I_q) : \chi \times \chi \rightarrow \mathcal{R}$$

for finite set $\chi = \{I_p | p \in \Omega\}$. It is easy to check that k KM energy (3.21) with kernel $k \equiv \mathcal{K}$ in (3.29) is equivalent to AA energy (3.26) with affinity A , up to a constant. Indeed, for any indicator $X \in \{0, 1\}^{|\Omega|}$ we have $X'X = \mathbf{1}'X$ implying

$$\frac{X'\mathcal{K}X}{\mathbf{1}'X} = \frac{X'AX}{2(\mathbf{1}'X)} + \frac{X'A'X}{2(\mathbf{1}'X)} + \delta \frac{X'X}{\mathbf{1}'X} = \frac{X'AX}{\mathbf{1}'X} + \delta.$$

Also, Section 3.4.1 uses eigen decomposition of \mathcal{K} to construct an explicit finite-dimensional Euclidean embedding⁵ $\phi_p \in \mathcal{R}^{|\Omega|}$ satisfying isometry (3.19) for any p.d. discrete kernel $k \equiv \mathcal{K}$. Minimizing k KM energy (3.17) over such embedding isometric to \mathcal{K} in (3.29) is equivalent to optimizing (3.21) and, therefore, (3.26).

Since average distortion energy (3.25) for arbitrary D is equivalent to average association for $A = -\frac{D}{2}$, it can also be converted to k KM with a proper kernel [197]. Using the corresponding kernel matrix (3.29) and (3.19) it is easy to derive Hilbertian distortion (metric) equivalent to original distortions D

$$\|I_p - I_q\|_k^2 := \frac{D + D'}{2} + 2\delta(\mathbf{1} \cdot \mathbf{1}' - \mathbf{I}). \quad (3.30)$$

For simplicity and without loss of generality, the rest of the paper assumes symmetric affinities $A = A'$ since non-symmetric ones can be equivalently replaced by $\frac{A+A'}{2}$. However, we do not assume positive definiteness and discuss diagonal shifts, if needed.

Weighted k KM and weighted AA: Weighted K-means [69] is a common extension of KM techniques incorporating some given point weights $w = \{w_p | p \in \Omega\}$. In the context of embedded points ϕ_p it corresponds to *weighted k KM* iteratively minimizing the weighted version of the mixed objective in (3.15)

$$F^w(S, m) := \sum_k \sum_{p \in S^k} w_p \|\phi_p - m_k\|^2. \quad (3.31)$$

Optimal segment centers m_k are now weighted means

$$\mu_{S^k}^w = \frac{\sum_{q \in S^k} w_q \phi_q}{\sum_{q \in S^k} w_q} \equiv \frac{\phi W S^k}{w' S^k} \quad (3.32)$$

⁵Mercer's theorem is a similar eigen decomposition for continuous p.d. kernels $k(x, y)$ giving infinite-dimensional Hilbert embedding $\phi(x)$. Discrete kernel embedding $\phi_p \equiv \phi(I_p)$ in Sec. 3.4.1 (3.55) has finite dimension $|\Omega|$, which is still much higher than the dimension of points I_p , *e.g.* \mathcal{R}^3 for colors. Sec. 3.4.1 also shows lower dimensional embeddings $\tilde{\phi}_p$ approximating isometry (3.19).

where the matrix formulation has weights represented by column vector $w \in \mathcal{R}^{|\Omega|}$ and diagonal matrix $W := \text{diag}(w)$. Assuming a finite dimensional data embedding $\phi_p \in \mathcal{R}^m$ this formulation uses *embedding matrix* $\phi := [\phi_p]$ with column vectors ϕ_p . This notation implies two simple identities used in (3.32)

$$\sum_{q \in S^k} w_q \equiv w' S^k \quad \text{and} \quad \sum_{q \in S^k} w_q \phi_q \equiv \phi W S^k. \quad (3.33)$$

Inserting weighted means (3.32) into mixed objective (3.31) produces a pairwise energy formulation for weighted k KM similar to (3.21)

$$F^w(S) := \sum_k \sum_{p \in S^k} w_p \|\phi_p - \mu_{S^k}^w\|^2 \quad (3.34)$$

$$\begin{aligned} &\stackrel{c}{=} - \sum_k \frac{\sum_{p,q \in S^k} w_p w_q \mathcal{K}_{pq}}{\sum_{p \in S^k} w_p} \\ &\equiv - \sum_k \frac{S^{k'} W \mathcal{K} W S^k}{w' S^k} \end{aligned} \quad (3.35)$$

where p.s.d kernel matrix $\mathcal{K} = \phi' \phi$ corresponds to the dot products in the embedding space, *i.e.* $\mathcal{K}_{pq} = \phi_p' \phi_q$.

Replacing the p.s.d. kernel with an arbitrary affinity matrix A defines a weighted AA objective generalizing (3.26) and (3.28)

$$E_{aa}^w(S) := - \sum_k \frac{S^{k'} W A W S^k}{w' S^k}. \quad (3.36)$$

Weighted AD can also be defined. Equivalence of k KM, AA, and AD in the general *weighted* case is discussed in [221, Appendix A].

Other kernel clustering criteria: Besides AA there are many other standard kernel clustering criteria defined by affinity matrices $A = [A_{pq}]$. For example, Average Cut (AC)

$$\begin{aligned} E_{ac}(S) &:= \sum_k \frac{\text{assoc}(S^k, \bar{S}^k)}{|S^k|} \equiv \sum_k \frac{S^{k'} A (\mathbf{1} - S^k)}{\mathbf{1}' S^k} \\ &= \sum_k \frac{S^{k'} (D - A) S^k}{\mathbf{1}' S^k} \end{aligned} \quad (3.37)$$

where $D := \text{diag}(d)$ is a *degree matrix* defined by node degrees vector $d := A \mathbf{1}$. The formulation on the last line (3.37) comes from the following identity valid for Boolean $X \in \{0, 1\}^{|\Omega|}$

$$X' D X = X' d.$$

Normalized Cut (NC) [211] in (3.38) is another well-known kernel clustering criterion. Due to popularity of NC we discuss it and its relation to other kernel clustering criteria in a dedicated Sec. 3.2.3.

Kernel selection issues: One of the practically important problems in kernel clustering is selection of the kernel or its bandwidth. It is known [158] that for a common class of kernels (*e.g.* popular Gaussian kernel), NC (3.40) and AC (3.37), AA (3.28) and k KM (3.21) have various *density biases*. In particular, AA and k KM with a small bandwidth isolate density modes [211] while AC and NC separate isolated data points [158]. Zelnik-Manor and Perona [256] discuss other related biases in NC. In practice the bandwidth choice is a trade-off between the prominence of the density biases for small bandwidths and lack of non-linear separation for large bandwidths. Instead of fitting a single bandwidth value, one can employ adaptive weights [158] or adaptive kernel bandwidths [256, 158], *e.g.* on K -nearest neighbor (KNN) graphs, to correct the density biases while keeping non-linearity of the decision boundary. Interestingly, in this case objectives NC, AC, k KM and AA become equivalent [158].

Pairwise vs. pointwise distortions

Equivalence of k KM to pairwise distortion criterion in (3.25) helps to juxtapose *kernel K-means* with *probabilistic K-means* (Sec.3.2.2) from one more point of view. Both methods generalize the basic K-means (3.6), (3.9) by replacing the Euclidean metric with a more general distortion measure $\|\cdot\|_d$. While pKM uses “pointwise” formulation (3.12) where $\|\cdot\|_d$ measures distortion between a point and a model, k KM uses “pairwise” formulation (3.20) where $\|\cdot\|_d = \|\cdot\|_k^2$ measures distortion between pairs of points.

These two different formulations are equivalent for Euclidean distortion (*i.e.* basic K-means), but the pairwise approach is strictly stronger than the pointwise version using the same Hilbertian distortion $\|\cdot\|_d = \|\cdot\|_k^2$ in non-Euclidean cases [221, Appendix B]. The corresponding pointwise approach is often called *weak kernel K-means*. Interestingly, weak k KM with standard Gaussian kernel can be seen as *K-modes* [202], see Fig. 3.3(g), which is closely related to popular *mean-shift* clustering [53], see [221, Appendix B]. An extended version of Tab. 3.3 including weighted KM and weak k KM is given in [221, Fig.34].

3.2.3 NC objective and its relation to AA, AC, and k KM

Sec. 3.2.2 has already discussed k KM and many related *kernel clustering* criteria based on specified affinities $A = [A_{pq}]$. This section is focused on a related kernel clustering

method, Normalized Cut (NC) [211]. Shi and Malik [211] also popularized kernel clustering optimization via *spectral relaxation*, which is different from iterative K-means algorithms (3.22) (3.23). Note that there are many other popular optimization methods for different clustering energies using pairwise affinities [55, 111, 238, 125, 105], which are outside the scope of this work.

The Normalized Cut (NC) objective [211] is defined as

$$\begin{aligned} E_{nc}(S) &:= -\sum_k \frac{assoc(S^k, S^k)}{assoc(\Omega, S^k)} \\ &\equiv -\sum_k \frac{S^{k'} AS^k}{\mathbf{1}' AS^k} \stackrel{c}{=} \sum_k \frac{S^{k'} A(\mathbf{1} - S^k)}{\mathbf{1}' AS^k} \end{aligned} \quad (3.38)$$

where *association* (3.27) is defined by a given *affinity matrix* A . The matrix formulations above shows that the difference between NC and AA (3.28) or AC (3.37) is in the normalization. In fact, normalization by $\mathbf{1}' AS^k$ is chosen specifically to make *normalized average association* equivalent to *normalized cut*, the last two expressions in (3.38). In contrast, AA and AC are distinct objectives normalized by $\mathbf{1}' S^k \equiv |S^k|$, which is k -th segment's size or cardinality. NC (3.38) normalizes by weighted cardinality. Indeed, using $d := A'\mathbf{1}$

$$\mathbf{1}' AS^k \equiv d' S^k \equiv \sum_{p \in S^k} d_p$$

where weights $d = \{d_p | p \in \Omega\}$ are *node degrees*

$$d_p := \sum_{q \in \Omega} A_{pq}. \quad (3.39)$$

For shortness, NC objective will be formatted like (3.28)

$$E_{nc}(S) \equiv -\sum_k \frac{S^{k'} AS^k}{d' S^k}. \quad (3.40)$$

It is known [158] that affinities such that $d_p \approx const$, *e.g.* on K -nearest neighbor (KNN) graphs, remove various *density biases* in kernel clustering. In this case objectives NC (3.40), AC (3.37), and AA (3.28) become equivalent. More generally, Bach & Jordan [9], Dhillon et al. [64] showed that NC objective can always be reduced to weighted AA or k KM with specific weights and affinities.

Our matrix notation makes equivalence between NC (3.40) and weighted AA (3.36) straightforward. Indeed, objective (3.40) with A coincides with (3.36) for weights w and affinity \tilde{A}

$$w = d = A'\mathbf{1} \quad \text{and} \quad \tilde{A} = W^{-1} A W^{-1}. \quad (3.41)$$

The weighted version of k KM procedure (3.23) [221, Appendix A] minimizes weighted AA (3.36) only for p.s.d. affinities, but positive definiteness of A is not critical. For example, an extension of the *diagonal shift* (3.29) [197] can convert NC (3.40) with arbitrary (symmetric) A to an equivalent NC objective with p.s.d. affinity

$$\mathcal{K} = A + \delta \cdot D \quad (3.42)$$

using *degree matrix* $D := \text{diag}(d) \equiv W$ and sufficiently large δ . Indeed, for indicators $X \in \{0, 1\}^{|\Omega|}$ we have $X'DX = d'X$ and

$$\frac{X'\mathcal{K}X}{d'X} = \frac{X'AX}{d'X} + \delta \frac{X'DX}{d'X} = \frac{X'AX}{d'X} + \delta.$$

Positive definite \mathcal{K} (3.42) implies positive definite affinity (3.41) of weighted AA

$$\tilde{\mathcal{K}} = D^{-1}\mathcal{K}D^{-1} = D^{-1}AD^{-1} + \delta D^{-1}. \quad (3.43)$$

The weighted version of k KM procedure (3.23) for this p.d. kernel [65] greedily optimizes NC objective (3.40) for any (symmetric) A .

3.2.4 Optimization methods for kernel clustering

As discussed in the previous section, both NC and AC can be reduced to weighted AA. Thus, all of these objectives can be optimized by basic kernel K-means procedure (1.18, 3.23) or its weighted variant. However, there are many other standard methods for approximate optimization of NP-hard kernel clustering energies.

Spectral relaxation: Shi, Malik, and Yu [211, 251] popularized *spectral relaxation* methods in the context of normalized cuts. Such methods also apply to AA and other problems [211]. For example, similarly to [251] one can rewrite AA energy (3.26) as

$$E_{aa}(S) = -\text{tr}(Z'AZ) \quad \text{for } Z := \left[\dots, \frac{S^k}{\sqrt{|S^k|}}, \dots \right]$$

where Z is a $|\Omega| \times K$ matrix of normalized indicator vectors S^k . Orthogonality $(S^i)'S^j = 0$ implies $Z'Z = I_K$ where I_K is an identity matrix of size $K \times K$. Minimization of the *trace* energy above with relaxed Z constrained to a “unit sphere” $Z'Z = I_K$ is a simple representative example of *spectral relaxation* in the context of AA. This relaxed trace optimization is a generalization of *Rayleigh quotient* problem that has an exact closed

form solution in terms of K largest eigenvectors for matrix A . This approach extends to general multi-label weighted AA and related graph clustering problems, *e.g.* AC and NC [211, 251]. The main computational difficulties for spectral relaxation methods are explicit eigen decomposition for large matrices and integrality gap - there is a final heuristics-based discretization step for extracting an integer solution for the original combinatorial problem from an optimal relaxed solution. For example, one basic discretization heuristic is to run K-means over the row-vectors of the optimal relaxed Z .

Fuzzy kernel k-means: Buhmann et al. [107, 197] address the general AD and AA energies via mean-field approximation. They derive an iterative algorithm that can be seen as a soft or *fuzzy* version⁶ of k KM procedure (3.23). In particular, at current segments S_t^k they compute unary “potentials”

$$U_{p,t}^k = \frac{S_t^{k'} \mathcal{K} S_t^k}{|S_t^k|^2} - 2 \frac{\mathbf{1}_p' \mathcal{K} S_t^k}{|S_t^k|} \quad (3.44)$$

where $U_{p,t}^k$ is a penalty for assigning label k to pixel p identical to the expression evaluated in (3.23). But, instead of updating point labels according to the lowest penalty $S_{p,t+1} = \arg \min_k U_{p,t}^k$ as in (3.23), the updates in [107] use *soft-min* operation based on *temperature* parameter T

$$S_{p,t+1}^k = \frac{\exp\left(\frac{-U_{p,t}^k}{T}\right)}{\sum_l \exp\left(\frac{-U_{p,t}^l}{T}\right)} \quad (3.45)$$

where soft assignments $S_p^k \in [0, 1]$ define probability distributions $(S_p^k | 1 \leq k \leq K) \in \Delta^K$ over labels, see the “alternative” side of Table 3.1. As $T \rightarrow 0$ soft-min (3.45) converges to binary indicators $S_p^k \in \{0, 1\}$ and distributions over labels become vertices of simplex Δ^K . That is, soft-min (3.45) reduces to “hard-min” $S_{p,t+1}^k = \arg \min_k U_{p,t}^k$ in (3.23).

Handling extra constraints: Some efforts to combine kernel clustering and regularization were made before us. For example, to combine k KM or NC objectives with Potts regularization, [132] normalizes the corresponding pairwise constraints by cluster sizes. This alters the Potts model to fit the problem to a standard trace-based formulation. In contrast, we address joint optimization via new bounds for the kernel clustering terms.

Adding non-homogeneous linear constraints into spectral relaxation techniques also requires approximations [252] or model modifications [245]. Exact optimization for the relaxed quadratic ratios (including NC) with arbitrary linear equality constraints is possible by solving a sequence of spectral problems [73]. To incorporate *must-link* and *cannot-link* constraints, [48] reformulated normalized cut and solved a different eigen problem .

⁶Similar to fuzzy K-means in [152, 69, 195] if extended to k KM.

3.3 Kernel Bounds

Our bound optimization approach allows to combine many standard kernel clustering objectives and any regularization terms with existing solvers. We interpret kernel clustering objectives as high-order energy terms and approximate them by linear upper bounds during optimization.

First, we review the general bound optimization principle and present basic K-means as an example. Sec. 3.3.2 derives kernel bounds for standard kernel clustering objectives. Without loss of generality, we assume symmetric affinities $A = A'$ since non-symmetric ones can be equivalently replaced by $\frac{A+A'}{2}$, *e.g.* see (3.29) in Sec.3.2.2. Positive definiteness of A is not assumed and *diagonal shifts* are discussed when needed. Move-making bound optimization for energy (3.1) is discussed in Sec. 3.3.3.

3.3.1 Bound optimization and K-means

As detailed in Sec. 1.4.1, bound optimization iteratively minimizes an auxiliary function and is guaranteed to decrease the objective. One of our motivation is that standard K-means algorithm can be interpreted as bound optimization, see Theorem 1.

Theorem 1 could be generalized to *probabilistic K-means* [114] by stating that block-coordinate descent for distortion clustering or ML model fitting (3.13) is a bound optimization [217, 218]. Theorem 1 can also be extended to pairwise and weighted versions of KM. For example, one straightforward extension is to show that $F^w(S, \mu_t^w)$ (3.31) with weighted means $\mu_t^w = \{\mu_{S_t^k}^w\}$ (3.32) is a bound for weighted KM objective $F^w(S)$ (3.34) [221, Th.6]. Then, some bound for pairwise *wkKM* energy (3.35) can also be derived [221, Cor.1]. It follows that bounds can be deduced for many kernel clustering criteria using their reductions to various forms of *kKM* reviewed in Sec.3.2.2 or 3.2.3.

Alternatively, the next Section 3.3.2 follows a more direct and intuitive approach to deriving kernel clustering bounds motivated by the following simple observation. Note that function $a_t(S)$ in Theorem 1 is *unary* with respect to S . Indeed, functions $F(S, m)$ (3.15) or $F^w(S, m)$ (3.31) can be written in the form

$$F(S, m) \equiv \sum_k \sum_p \|\phi_p - m_k\|^2 S_p^k \quad (3.46)$$

$$F^w(S, m) \equiv \sum_k \sum_p w_p \|\phi_p - m_k\|^2 S_p^k \quad (3.47)$$

highlighting the sum of unary terms for variables S_p^k . Thus, bounds for KM or weighted KM objectives are *modular* (linear) function of S . This simple technical fact has useful implications that were previously overlooked. For example,

- in the context of bound optimization, KM can be integrated with many regularization potentials whose existing solvers can work with extra unary (linear) terms
- assuming real-valued relaxation of indicators S^k , linearity of upper bound $a_t(S)$ (1.20) implies that the bounded function $F(S) \in \mathcal{C}^1$ (3.21) is *concave*, see Fig. 1.11.

In Section 3.3.2 we confirm that many standard kernel clustering objectives in Sections 3.2.2 and 3.2.3 have *concave relaxations*. Thus, their linear upper bounds easily follow from the corresponding first-order Taylor expansions, see Figure 1.11 and Table 3.4.

3.3.2 Kernel Bounds for AA, AC, and NC

The next lemma helps to find linear bounds for clustering terms AA, AC, or NC in Tab. 3.4 (Theorem 2) and bounds for our joint energy (3.1) in Corolary 2.

Lemma 1 (concave relaxation). *Consider function $\widehat{e} : \mathcal{R}^\Omega \rightarrow \mathcal{R}^1$ defined by matrix \mathcal{K} and vector w as*

$$\widehat{e}(X) := -\frac{X' \mathcal{K} X}{w' X}. \quad (3.48)$$

Function $\widehat{e}(X)$ is concave over region $w' X > 0$ assuming (symmetric) matrix \mathcal{K} is positive semi-definite (see Fig. 3.6).

Proof. Omitted here, see [223] for detail. □

The first-order Taylor expansion at current solution X_t

$$T_t(X) := \widehat{e}(X_t) + \nabla \widehat{e}(X_t)' (X - X_t)$$

is a bound for the concave function $\widehat{e}(X)$ (3.48). Its gradient⁷

$$\nabla \widehat{e}(X_t) = w \frac{X_t' \mathcal{K} X_t}{(w' X_t)^2} - \mathcal{K} X_t \frac{2}{w' X_t} \quad (3.49)$$

gives linear bound $T_t(X)$ for concave function $\widehat{e}(X)$ at X_t

$$T_t(X) \equiv \nabla \widehat{e}(X_t)' X. \quad (3.50)$$

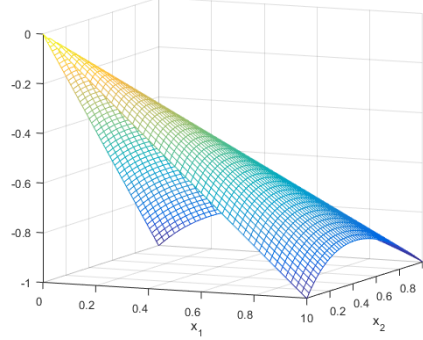


Figure 3.6: Example: concave function $\hat{e}(X) = -\frac{X'X}{\mathbf{1}'X}$ for $X \in [0, 1]^2$. Note that convexity/concavity of similar rational functions with quadratic enumerator and linear denominator is known in other optimization areas, *e.g.* [27, p.72] states convexity of $\frac{x^2}{y}$ for $y > 0$ and [12, exercise 3.14] states convexity of $\frac{(v'X)^2}{w'X}$ for $w'X > 0$.

As shown in the second column of Table 3.4, common kernel clustering objectives defined by affinity matrix A such as AA (3.28), AC (3.37), and NC (3.40) have the form

$$E_A(S) = \sum_k e(S^k)$$

with function $e(X)$ as in (3.48) from Lemma 1. However, arbitrary affinity A may not correspond to a positive semi-definite \mathcal{K} in (3.48) and $e(X)$ may not be concave for $X \in \mathcal{R}^{|\Omega|}$. However, the *diagonal shift* trick [197] in (3.29) works here too. The third column in Table 3.4 shows concave function $\hat{e}(X)$ that equals $e(X)$ for any non-zero Boolean $X \in \{0, 1\}^{|\Omega|}$, up to a constant. Indeed, for AA

$$\hat{e}(X) = -\frac{X'(\delta\mathbf{I} + A)X}{\mathbf{1}'X} = -\frac{X'AX}{\mathbf{1}'X} - \delta \stackrel{c}{=} e(X)$$

since $X'X = \mathbf{1}'X$ for Boolean X . Clearly, $\delta\mathbf{I} + A$ is p.s.d. for sufficiently large δ ⁸ and Lemma 1 implies that the first-order Taylor expansion $T_t(X)$ (3.50) is a linear bound for concave function $\hat{e}(X)$. Equivalence between e and \hat{e} over Booleans allows to use $T_t(X)$ as a bound for e when optimizing over indicators X . Function $\hat{e} : \mathcal{R}^{|\Omega|} \rightarrow \mathcal{R}^1$ can be described as a *concave relaxation* of the high-order pseudo-boolean function $e : \{0, 1\}^{|\Omega|} \rightarrow \mathcal{R}^1$.

⁷Function \hat{e} and gradient $\nabla \hat{e}$ are defined only at non-zero indicators X_t where $w'X_t > 0$. We can formally extend \hat{e} to $X = \mathbf{0}$ and make the bound T_t work for \hat{e} at $X_t = \mathbf{0}$ with some *supergradient*. However, $X_t = 0$ is not a problem in practice since it corresponds to an empty segment.

⁸A sufficiently large δ can be determined by the smallest eigenvalue of A .

Concave relaxation \hat{e} for AC in Table 3.4 follows from the same diagonal shift $\delta\mathbf{I}$ as above. But NC requires diagonal shift δD with degree matrix $D = \text{diag}(d)$ as in (3.42). Indeed,

$$\hat{e}(X) = -\frac{X'(\delta D + A)X}{d'X} = -\frac{X'AX}{d'X} - \delta \stackrel{c}{=} e(X) \quad (3.51)$$

since $X'DX \equiv X'\text{diag}(d)X = d'X$ for any Boolean X . Clearly, $\delta D + A$ is p.s.d. for sufficiently large δ assuming $d_p > 0$ for all $p \in \Omega$. Concave relaxations and the corresponding Taylor-based bounds for $E_A(S)$ in Table 3.4 imply the following theorem.

Theorem 2 (kernel bound for E_A). *For (symmetric) affinity matrix A and current solution S_t the following is a unary (linear) bound for any kernel clustering energy $E_A(S)$ in Tab. 3.4*

$$a_t(S) = \sum_k \nabla \hat{e}(S_t^k)' S^k \quad (3.52)$$

where \hat{e} and $\nabla \hat{e}$ are defined in (3.48), (3.49) and δ is large enough so that the corresponding \mathcal{K} in Tab. 3.4 is positive semi-definite.

objective $E_A(S)$	formulation $e(X)$ in $\sum_k e(S^k)$	concave relaxation $\hat{e}(X)$ (3.48)	\mathcal{K} and w in Lemma 1
AA (3.28)	$-\frac{X'AX}{\mathbf{1}'X}$	$-\frac{X'(\delta\mathbf{I}+A)X}{\mathbf{1}'X}$	$\mathcal{K} = \delta\mathbf{I} + A, w = \mathbf{1}$
AC (3.37)	$\frac{X'(D-A)X}{\mathbf{1}'X}$	$-\frac{X'(\delta\mathbf{I}+A-D)X}{\mathbf{1}'X}$	$\mathcal{K} = \delta\mathbf{I} + A - D, w = \mathbf{1}$
NC (3.40)	$-\frac{X'AX}{d'X}$	$-\frac{X'(\delta D+A)X}{d'X}$	$\mathcal{K} = \delta D + A, w = d$

Table 3.4: *Kernel bounds* for different kernel clustering objectives $E_A(S)$ can be derived from (3.49) and (3.52) using corresponding \mathcal{K} and w . The second column shows formulations of these objectives $E_A(S) \equiv \sum_k e(S^k)$ using functions e over segment indicator vectors $S^k \in \{0, 1\}^{|\Omega|}$. (3.52) gives a unary (linear) upper bound for $E_A(S)$ at S_t based on the first-order Taylor approximation of *concave relaxation* function $\hat{e} : \mathcal{R}^\Omega \rightarrow \mathcal{R}^1$ (3.48).

Note that though Theorem 2 requires a positive semi-definite matrix \mathcal{K} , we found in practice that iterative optimization of (3.52) for non-PSD matrix may still decrease the energy. An example is with standard KNN kernel that is not necessarily PSD.

Similarly to Theorem 1, optimization of our linear kernel bound in Theorem 2 can be related to k KM updates (3.23). Indeed, (3.52) can be written in the form

$$a_t(S) \equiv \sum_k \sum_p \mathbf{1}'_p \nabla \widehat{e}(S_t^k) S_p^k = \sum_p \left(\sum_k \mathbf{1}'_p \nabla \widehat{e}(S_t^k) S_p^k \right)$$

that breaks into the sum of linear terms for each p

$$\sum_k \mathbf{1}'_p \nabla \widehat{e}(S_t^k) S_p^k. \quad (3.53)$$

Each of these can be optimized independently over probability simplex $\sum_k S_p^k = 1$. The optimal solution for (3.53) is always at one of K corners of the simplex corresponding to label k with the lowest potential $\mathbf{1}'_p \nabla \widehat{e}(S_t^k)$. For example, assuming AA objective (3.28) with $w = \mathbf{1}$ and $A = \mathcal{K}$, then (3.49) implies optimal k as in the “hard” k KM update (3.23). Interestingly, combining (3.53) with an “entropy barrier” pushing the solution away from the simplex corners

$$\sum_k \mathbf{1}'_p \nabla \widehat{e}(S_t^k) S_p^k + T \cdot \sum_k S_p^k \log S_p^k$$

results in the optimal “soft” k KM update (3.45) as in [107]. In their mean-field approach, the objective above comes as KL divergence between Gibbs distributions for the exact and approximate AA energies. Note $\mathbf{1}'_p \nabla \widehat{e}(S_t^k) \equiv U_{p,t}^k$, see (3.44).

For the joint energy (3.1) combining kernel clustering and regularization terms we can use the following bounds.

Corollary 2 (kernel bound for (3.1)). *For any (symmetric) affinity matrix A and any current solution S_t the following is an auxiliary function for energy (3.1) with any clustering term $E_A(S)$ from Tab. 3.4*

$$a_t(S) = \sum_k \nabla \widehat{e}(S_t^k)' S^k + \gamma \sum_{c \in \mathcal{F}} E_c(S_c) \quad (3.54)$$

where \widehat{e} and $\nabla \widehat{e}$ are defined in (3.48), (3.49) and δ is large enough so that the corresponding \mathcal{K} in Tab. 3.4 is positive semi-definite.

Algorithm 3: α -Expansion for Kernel Cut

Input : Affinity matrix \mathcal{A} of size $|\Omega| \times |\Omega|$;
Initial labeling S_0^1, \dots, S_0^K

Output: S^1, \dots, S^K : partition of the set Ω

```
1 Find p.s.d. matrix  $\mathcal{K}$  as in Table 3.4. Set  $t := 0$ ;  
2 while not converged do  
3   Set  $a_t(S)$  to be kernel bound (3.54) at current partition  $S_t$ ;  
4   for each label  $\alpha \in \mathcal{L} = \{1, \dots, K\}$  do  
5     Find  $S_t := \arg \min a_t(S)$  within one  $\alpha$  expansion of  $S_t$ ;  
6   end  
7   Set  $t := t + 1$ ;  
8 end
```

3.3.3 Move-making algorithms

Combination (3.54) of regularization potentials with a unary (linear) bound $\sum_k \nabla \hat{e}(S_t^k)' S^k$ for high-order term $E_A(S)$ can be optimized with many standard discrete or continuous multi-label methods including graph cuts [31, 109], message passing [124], LP relaxations [240], or well-known continuous convex formulations [39, 41, 57]. We focus on MRF regularizers (see Sec.3.2.1) commonly addressed by graph cuts [31]. We discuss some details of kernel bound optimization technique using such methods.

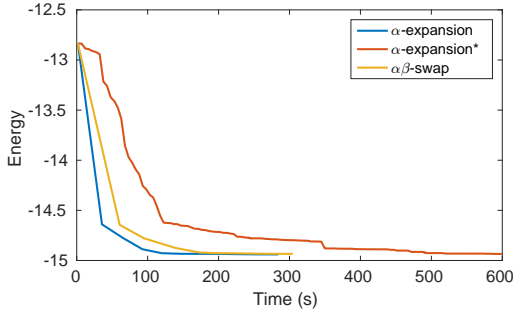
Step I of the bound optimization algorithm (Fig. 1.10) using auxiliary function $a_t(S)$ (3.54) for energy $E(S)$ (3.1) with regularization potentials reviewed in Sec.3.2.1 can be done via move-making methods [31, 120, 60]. Step II requires re-evaluation of the first term in (3.54), *i.e.* the kernel bound for E_A . Estimation of gradients $\nabla \hat{e}(S_t^k)$ in (3.49) has complexity $O(K|\Omega|^2)$.

Even though the global optimum of a_t at step I (Fig. 1.10) is not guaranteed for general potentials E_c , it suffices to decrease the bound in order to decrease the energy, *i.e.* (1.14a) and (1.14b) imply

$$a_t(S_{t+1}) \leq a_t(S_t) \quad \Rightarrow \quad E(S_{t+1}) \leq E(S_t).$$

For example, Algorithm 3 shows a version of our kernel cut algorithm using α -expansion [31] for decreasing bound $a_t(S)$ in (3.54). Other moves are also possible, for example $\alpha\beta$ -swap.

In general, *tighter* bounds work better. Thus, we do not run iterative move-making algorithms for bound a_t until convergence before re-estimating a_{t+1} . Instead, one can



(a) Versions of Kernel Cut

Compare	against	# of wins	p-value [†]
α -expansion	$\alpha\beta$ -swap	135/200	10^{-6}
α -expansion	α -expansion*	182/200 [‡]	10^{-34} [‡]

[†] The probability to exceed the given number of wins by random chance.

[‡] The algorithm stopped due to time limit (may cause incorrect number of wins).

(b) BSDS500 training dataset

Figure 3.7: Typical energy evolution wrt different moves and frequency of bound updates. α -expansion updates the bound after a round of expansions, α -expansion* updates the bound after each expansion move. Initialization is a regular 5×5 grid of patches.

reestimate the bound either after each move or after a certain number of moves. One should decide the order of iterative move making and bound evaluation. In the case of α -expansion, there are at least three options: updating the bound after a single expansion step, or after a single expansion loop, or after the convergence of α -expansion. More frequent bound recalculation slows down the algorithm, but makes the bound tighter. The particular choice generally depends on the trade-off between the speed and solution quality. However, in our experiments more frequent update does not always improve the energy, see Fig. 3.7. We recommend updating the bound after a single loop of expansions, see Alg.3. We also evaluated a swap move version of our kernel cut method with bound re-estimation after a complete $\alpha\beta$ -swaps loop, see Fig. 3.7.

3.4 Data Embeddings and Spectral Bounds

This section shows a different bound optimization approach to kernel clustering. In contrast to the bounds explicitly using affinity A or kernel matrices \mathcal{K} in Sec. 3.3.2, the new approach is based on explicit use of isometric data embeddings ϕ , see Sec. 3.2.2. While the general Mercer theorem guarantees existence of such possibly infinite dimensional Hilbert space embedding, we show finite dimensional Euclidean embedding

$$\phi := [\phi_p] \quad \text{where} \quad \{\phi_p | p \in \Omega\} \subset \mathcal{R}^{|\Omega|}$$

with exact isometry (3.18, 3.19) to kernels \mathcal{K} in Table 3.4 and lower dimensional embeddings

$$\tilde{\phi} := [\tilde{\phi}_p] \quad \text{where} \quad \{\tilde{\phi}_p | p \in \Omega\} \subset \mathcal{R}^m \quad \text{for} \quad m \leq |\Omega|$$

that can approximate the same isometry with any accuracy. The embeddings use eigen decompositions of the kernels.

Explicit embeddings allow to formulate exact or approximate *spectral bounds* for standard kernel clustering objectives like AA, AC, NC. This approach is very closely related to spectral relaxation, see Sec. 3.4.3. For example, optimization of our approximate spectral bounds for $m = K$ is similar to standard discretization heuristics using K-means over eigenvectors [211]. Our bound optimization framework provides justification for such heuristics. Moreover, our spectral bounds also allow to optimize joint energy (3.1) combining kernel clustering objectives with common regularization terms.

Spectral bound is a useful alternative to kernel bound in Sec. 3.3.2. Their complexity and other numerical properties are different. In particular, spectral bound optimization with lower dimensional Euclidean embeddings $\tilde{\phi}$ for $m \ll |\Omega|$ is often less sensitive to local minima. This may lead to better solutions, even though such embeddings $\tilde{\phi}$ are only approximately isometric to given pairwise affinities. For $m = |\Omega|$, the spectral bound is mathematically equivalent to the kernel bound, but their numerical representations are different. Fig. 3.8 summarizes the relationship between our (*kernel* and *spectral*) bounds for kernel clustering objective $E_A(S)$.

3.4.1 Exact and approximate embeddings ϕ for k KM

This section uses some standard methodology [56] to build the finite-dimensional embedding $\phi_p \equiv \phi(I_p)$ with exact or approximate isometry (3.18, 3.19) to any given positive definite kernel k over finite data set $\{I_p | p \in \Omega\}$. As discussed in Sec. 3.2.2, k KM and other kernel

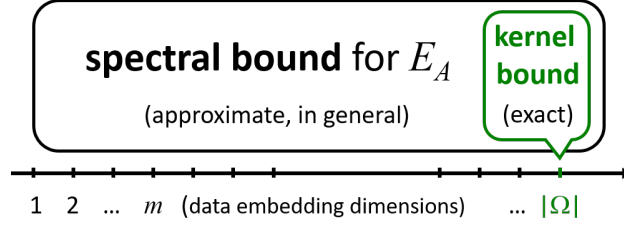


Figure 3.8: Interpreting our linear bounds for E_A term in (3.1) via K -means: optimization of the *spectral* bound (alone) is equivalent to K -means algorithm over approximately isometric data embeddings in \mathcal{R}^m for $m \leq |\Omega|$, see Sec. 3.4. As m approaches $|\Omega|$, the isometry becomes more accurate and our approximate spectral bound for E_A reduces to the exact *kernel* bound. While relations between E_A and K -means were known for $m = K$ [211] (as a heuristic, see Sec.3.4.3) and $m = |\Omega|$ [197, 9, 64] (as energy equivalence), we establish it in a new bound optimization context essential for our work.

clustering methods are typically defined by affinities/kernels k and energy (3.21) rather than by high-dimensional embeddings ϕ with basic KM formulation (3.17). Nevertheless, data embeddings ϕ_p could be useful and some clustering techniques explicitly construct them [211, 171, 197, 14, 9, 253]. In particular, if dimensionality of the embedding space is relatively low then the basic iterative KM procedure (3.22) minimizing (3.17) could be more efficient than its kernel variant (3.23) for quadratic formulation (3.21). Even when working with a given kernel k it may be algorithmically beneficial to build the corresponding isometric embedding ϕ . Below we discuss finite-dimensional Euclidean embeddings in \mathcal{R}^m ($m \leq |\Omega|$) allowing to approximate standard kernel clustering via basic KM.

First, we show an exact Euclidean embedding isometric to a given kernel. Any finite data set $\{I_p | p \in \Omega\}$ and any given kernel k define a positive definite *kernel matrix*⁹

$$\mathcal{K}_{pq} = k(I_p, I_q)$$

of size $|\Omega| \times |\Omega|$. The *eigen decomposition* of this matrix

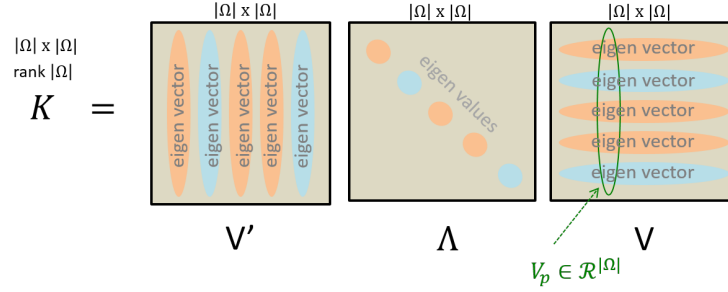
$$\mathcal{K} = V' \Lambda V$$

involves diagonal matrix Λ with non-negative eigenvalues and orthogonal matrix V whose rows are eigenvectors, see Fig. 3.9(a). Non-negativity of the eigenvalues is important for

⁹If k is given as a continuous kernel $k(x, y) : \mathcal{R}^N \times \mathcal{R}^N \rightarrow \mathcal{R}$ matrix \mathcal{K} is its *restriction* to finite data set $\{I_p | p \in \Omega\} \subset \mathcal{R}^N$.



(a) decomposition $\mathcal{K} = V'\Lambda V$



(b) decomposition $\tilde{\mathcal{K}} = (V^m)'\Lambda^m V^m$ for $m < |\Omega|$

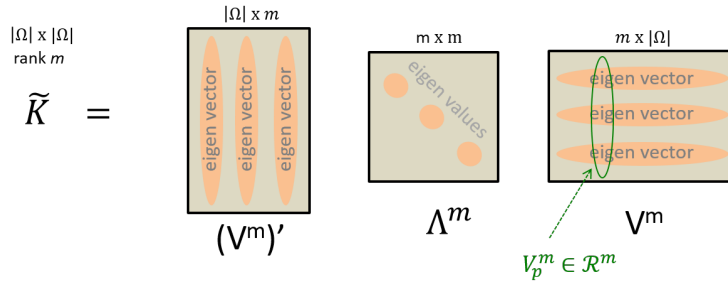


Figure 3.9: Eigen decompositions for kernel matrix \mathcal{K} (a) and its rank m approximation $\tilde{\mathcal{K}}$ (b) minimizing Frobenius errors (3.56) [56]. Decompositions (a,b) give explicit embeddings (3.55,3.58) isometric to the kernels, as in the Mercer theorem. One specific example for the Gaussian kernel is in Fig. 3.10.

obtaining decomposition $\Lambda = \sqrt{\Lambda} \cdot \sqrt{\Lambda}$ allowing us to define the following Euclidean space embedding

$$\phi_p := \sqrt{\Lambda} V_p \in \mathcal{R}^{|\Omega|} \quad (3.55)$$

where V_p are column of V , see Fig. 3.9(a). This embedding satisfies isometry (3.18,3.19) since

$$\langle \phi_p, \phi_q \rangle = (\sqrt{\Lambda} V_p)' (\sqrt{\Lambda} V_q) = \mathcal{K}_{pq} = k(I_p, I_q).$$

Note that (3.55) defines a simple finite dimensional embedding $\phi_p \equiv \phi(I_p)$ only for subset of points $\{I_p | p \in \Omega\}$ in \mathcal{R}^N based on a discrete kernel, *i.e.* matrix \mathcal{K}_{pq} . In contrast,

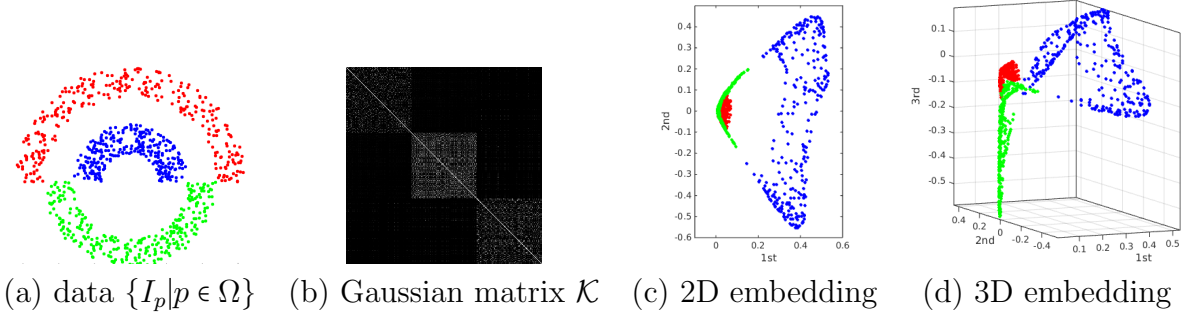


Figure 3.10: Low-dimensional Euclidean embeddings (3.58) for $m = 2$ and $m = 3$ in (c,d) are approximately isometric to a given affinity matrix (b) over the data points in (a). The approximation error (3.57) decreases for larger m . While generated by standard MDS methodology [56], it is intuitive to call embeddings ϕ in (3.55) and (3.58) as (exact or approximate) *isometry eigenmap* or *eigen isomap*.

Mercer's theorem should produce a more general infinite dimensional Hilbert embedding $\phi(x)$ for any $x \in \mathcal{R}^N$ by extending the eigen decomposition to continuous kernels $k(x, y)$. In either case, however, the embedding space dimensionality is much higher than the original data space. For example, ϕ_p in (3.55) has dimension $|\Omega|$, which is much larger than the dimension of data I_p , e.g. 3 for RGB colors.

Embedding (3.55) satisfying isometry (3.18,3.19) is not unique. For example, any decomposition $\mathcal{K} = G'G$, e.g. Cholesky [88], defines a mapping $\phi_p^G := G_p$ with desired properties. Also, rotational matrices R generate a class of isometric embeddings $\phi_p^R := R\phi_p$.

It is easy to build lower dimensional embeddings by weakening the exact isometry requirements (3.18,3.19) following the standard *multi-dimensional scaling* (MDS) methodology [56], as detailed below. Consider a given rank $m < |\Omega|$ approximation $\tilde{\mathcal{K}}$ for kernel matrix \mathcal{K} minimizing Frobenius norm errors [56]

$$\|\mathcal{K} - \tilde{\mathcal{K}}\|_F := \sum_{pq \in \Omega} (\mathcal{K}_{pq} - \tilde{\mathcal{K}}_{pq})^2. \quad (3.56)$$

It is well known [56, 88] that the minimum Frobenius error is achieved by

$$\tilde{\mathcal{K}} = (V^m)' \Lambda^m V^m$$

where V^m is a submatrix of V including m rows corresponding to the largest m eigenvalues of \mathcal{K} and Λ^m is the diagonal matrix of these eigenvalues, see Fig. 3.9(b). The corresponding

minimum Frobenius error is given by the norm of zeroed out eigenvalues

$$\|\mathcal{K} - \tilde{\mathcal{K}}\|_F = \sqrt{\lambda_{m+1}^2 + \dots + \lambda_{|\Omega|}^2}. \quad (3.57)$$

It is easy to check that lower dimensional embedding

$$\tilde{\phi}_p := \sqrt{\Lambda^m} V_p^m \in \mathcal{R}^m \quad (3.58)$$

is isometric with respect to approximating kernel $\tilde{\mathcal{K}}$, that is

$$\langle \tilde{\phi}_p, \tilde{\phi}_q \rangle = \tilde{\mathcal{K}}_{pq} \approx \mathcal{K}_{pq}. \quad (3.59)$$

Fig. 3.10 shows examples of low-dimensional approximate isometry embeddings (3.58) for a Gaussian kernel. Note that $\tilde{\phi}_p \in \mathcal{R}^m$ (3.58) can be obtained from $\phi_p \in \mathcal{R}^{|\Omega|}$ (3.55) by selecting coordinates corresponding to dimensions of the largest m eigenvalues.

According to (3.57) lower dimensional embedding $\tilde{\phi}_p$ in (3.58) is nearly-isometric to kernel matrix \mathcal{K} if the ignored dimensions have sufficiently small eigenvalues. Then (3.58) may allow efficient approximation of kernel K-means. For example, if sufficiently many eigenvalues are close to zero then a small rank m approximation $\hat{\mathcal{K}}$ will be sufficiently accurate. In this case, we can use a basic iterative K-means procedure directly in \mathcal{R}^m with $O(|\Omega|m)$ complexity of each iteration. In contrast, each iteration of the standard kernel K-means (3.21) is $O(|\Omega|^2)$ in general¹⁰.

There is a different way to justify approximate low-dimensional embedding $\tilde{\phi}_p$ ignoring small eigenvalue dimensions in ϕ_p . The objective in (3.21) for exact kernel \mathcal{K} is equivalent to the basic K-means (3.15) over points ϕ_p (3.55). The latter can be shown to be equivalent to (probabilistic) K-means (3.12) over columns V_p in orthonormal matrix V using weighted distortion measure

$$\|V_p - \mu\|_\Lambda^2 := \sum_{i=1}^{|\Omega|} \lambda_i (V_p[i] - \mu[i])^2 = \|\phi_p - \sqrt{\Lambda}\mu\|^2$$

where index $[i]$ specifies coordinates of the column vectors. Thus, a good approximation is achieved when ignoring coordinates for small enough eigenvalues contributing low weight in the distortion above. This is equivalent to K-means (3.15) over points (3.58).

¹⁰Without *KNN* or other special kernel accelerations.

3.4.2 Spectral Bounds for AA, AC, and NC

The last Section showed that k KM clustering with given p.s.d. kernel \mathcal{K} can be approximated by basic KM over low-dimensional Euclidean embedding $\tilde{\phi} \in \mathcal{R}^m$ (3.58) with approximate isometry to \mathcal{K} (3.59). Below we use equivalence of standard kernel clustering criteria to k KM, as discussed in Sections 3.2.2 and 3.2.3, to derive the corresponding low-dimensional embeddings for AA, AC, NC. Then, equivalence of KM to bound optimization (Theorem 1) allows to formulate our approximate *spectral bounds* for the kernel clustering and joint energy (3.1). The results of this Section are summarized in Tab. 3.5. For simplicity, assume symmetric affinity matrix A . If not, equivalently replace A by $\frac{A+A'}{2}$.

Average association (AA): Diagonal shift $\mathcal{K} = \delta\mathbf{I} + A$ in (3.29) converts AA (3.28) with A to equivalent k KM (3.21) with p.d. kernel \mathcal{K} . In practice, sufficiently large δ is added by finding the minimum eigenvalue of A . We seek rank- m approximation $\tilde{\mathcal{K}}$ minimizing Frobenius error $\|\mathcal{K} - \tilde{\mathcal{K}}\|_F$. Provided eigen decomposition $A = V'\Lambda V$, equation (3.58) gives low-dimensional embedding (also in Tab. 3.5)

$$\tilde{\phi}_p = \sqrt{\delta\mathbf{I}^m + \Lambda^m} V_p^m \quad (3.60)$$

corresponding to optimal approximation kernel $\tilde{\mathcal{K}}$. It follows that KM (3.22) over this embedding approximates AA objective (3.21). Note that the eigenvectors (rows of matrix V , Fig. 3.9) also solve the spectral relaxation for AA in Tab. 3.6. However, ad hoc discretization by KM over points V_p^K may differ from the result for points (3.60).

Average cut (AC): As follows from objective (3.37) and diagonal shift (3.29) [197], average cut clustering for affinity A is equivalent to minimizing k KM objective with kernel $\mathcal{K} = \delta\mathbf{I} + A - D$ where D is a diagonal matrix of node degrees $d_p = \sum_q A_{pq}$. Diagonal shift $\delta\mathbf{I}$ is needed to guarantee positive definiteness of the kernel. Eigen decomposition for $D - A = V'\Lambda V$ implies $\mathcal{K} = V'(\delta\mathbf{I} - \Lambda)V$. Then, (3.58) implies rank- m approximate isometry embedding (also in Tab. 3.5)

$$\tilde{\phi}_p = \sqrt{\delta\mathbf{I}^m - \Lambda^m} V_p^m \quad (3.61)$$

using the same eigenvectors (rows of V) that solve AC's spectral relaxation in Tab. 3.6. However, standard discretization heuristic using KM over $\tilde{\phi}_p = V_p^K$ may differ from the results for our approximate isometry embedding $\tilde{\phi}_p$ (3.61) due to different weighting.

Normalized cut (NC): According to [64] and a simple derivation in Sec.3.2.3 normalized cut for affinity A is equivalent to weighted k KM with kernel $\mathcal{K} = \delta D^{-1} + D^{-1} A D^{-1}$ (3.43) and node weights $w_p = d_p$ based on their degree. Weighted k KM (3.35) can be interpreted as KM in the embedding space with weights w_p for each point ϕ_p as in (3.31,3.32). The only

issue is computing m -dimensional embeddings approximately isometric to \mathcal{K} . Note that previously discussed solution $\tilde{\phi}$ in (3.58) uses eigen decomposition of matrix \mathcal{K} to minimize the sum of quadratic errors between \mathcal{K}_{pq} and approximating kernel $\tilde{\mathcal{K}}_{pq} = \langle \tilde{\phi}_p, \tilde{\phi}_q \rangle$. This solution may still be acceptable, but in the context of weighted points it seems natural to minimize an alternative approximation measure taking w_p into account. For example, we can find rank- m approximate affinity matrix $\tilde{\mathcal{K}}$ minimizing the sum of weighted squared errors

$$\sum_{pq \in \Omega} w_p w_q (\mathcal{K}_{pq} - \tilde{\mathcal{K}}_{pq})^2 = \|D^{\frac{1}{2}}(\mathcal{K} - \tilde{\mathcal{K}})D^{\frac{1}{2}}\|_F. \quad (3.62)$$

Substituting $\mathcal{K} = \delta D^{-1} + D^{-1}AD^{-1}$ gives an equivalent objective

$$\|D^{-\frac{1}{2}}(\delta D + A)D^{-\frac{1}{2}} - D^{\frac{1}{2}}\tilde{\mathcal{K}}D^{\frac{1}{2}}\|_F.$$

Consider rank- m matrix $\tilde{M} := D^{\frac{1}{2}}\tilde{\mathcal{K}}D^{\frac{1}{2}}$ as a new minimization variable. Its optimal value $(V^m)'(\delta \mathbf{I}^m + \Lambda^m)V^m$ follows from $D^{-\frac{1}{2}}(\delta D + A)D^{-\frac{1}{2}} = V'(\delta \mathbf{I} + \Lambda)V$ for eigen decomposition

$$D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \equiv V'\Lambda V. \quad (3.63)$$

Thus, optimal rank- m approximation kernel $\tilde{\mathcal{K}}$ is

$$\tilde{\mathcal{K}} = D^{-\frac{1}{2}}(V^m)'(\delta \mathbf{I}^m + \Lambda^m)V^m D^{-\frac{1}{2}}. \quad (3.64)$$

It is easy to check that m -dimensional embedding (also in Tab. 3.5)

$$\tilde{\phi}_p = \sqrt{\frac{\delta \mathbf{I}^m + \Lambda^m}{d_p}} V_p^m \quad (3.65)$$

is isometric to kernel $\tilde{\mathcal{K}}$, that is $\langle \tilde{\phi}_p, \tilde{\phi}_q \rangle = \tilde{\mathcal{K}}_{pq}$. Therefore, weighted KM (3.31) over low-dimensional embedding $\tilde{\phi}_p$ (3.65) with weights $w_p = d_p$ approximates NC objective (3.40).

Summary: The ideas above can be summarized as follows. Assume AA, AC, or NC objectives $E_A(S)$ with (symmetric) A . The third column in Table 3.5 shows kernels \mathcal{K} for equivalent k KM objectives $F(S)$ (3.21, 3.35). Following *eigenmap* approach (Fig. 3.9), we find rank- m approximate kernel $\tilde{\mathcal{K}} \approx \mathcal{K}$ minimizing Frobenius error $\|\tilde{\mathcal{K}} - \mathcal{K}\|_F$ (3.56) or its weighted version (3.62) and deduce embeddings $\tilde{\phi}_p \in \mathcal{R}^m$ (3.60), (3.61), (3.65) satisfying isometry

$$\tilde{\phi}_p' \tilde{\phi}_q = \tilde{\mathcal{K}}_{pq} \approx \mathcal{K}_{pq}.$$

objective $E_A(S)$	formulation $e(X)$ in $\Sigma_k e(S^k)$	equivalent k KM (3.21, 3.35) as in [197, 65]	eigen decomposition $V' \Lambda V = \dots$	embedding in \mathcal{R}^m , $m \leq \Omega $ with approx. isometry (3.59)	spectral bound for $E_A(S)$ at S_t
AA (3.28)	$-\frac{X'AX}{\mathbf{1}'X}$	$\mathcal{K} = \delta \mathbf{I} + A$	A	$\tilde{\phi}_p = \sqrt{\delta \mathbf{I}^m + \Lambda^m} V_p^m$ (3.60)	
AC (3.37)	$\frac{X'(D-A)X}{\mathbf{1}'X}$	$\mathcal{K} = \delta \mathbf{I} + A - D$	$D - A$	$\tilde{\phi}_p = \sqrt{\delta \mathbf{I}^m - \Lambda^m} V_p^m$ (3.61)	$F(S, \mu_t)$ (3.46) for points $\tilde{\phi}_p$
NC (3.40)	$-\frac{X'AX}{d'X}$	$\mathcal{K} = \delta D^{-1} + D^{-1} A D^{-1}$ weighted, $w_p = d_p$	$D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$	$\tilde{\phi}_p = \sqrt{\delta \mathbf{I}^m + \Lambda^m} V_p^m$ (3.65)	$F^w(S, \mu_t^w)$ (3.47) for points $\tilde{\phi}_p$

Table 3.5: *Spectral bounds* for objectives $E_A(S)$. The third column shows p.d. kernel matrices \mathcal{K} for the equivalent k KM energy (3.21). Eigen decomposition for matrices in the forth column defines our Euclidean embedding $\tilde{\phi}_p \in \mathcal{R}^m$ (fifth column) isometric to \mathcal{K} (3.59). Thus, \mathcal{K} -means over $\tilde{\phi}_p$ approximates k KM (3.21). Bounds for KM (last column) follow from Th. 1 where $\mu_t = \{\mu_{S_t^k}\}$ are means (3.16) and $\mu_t^w = \{\mu_{S_t^k}^w\}$ are weighted means (3.32). Functions $F(S, m)$ and $F^w(S, m)$ are modular (linear) w.r.t. S , see (3.46, 3.47).

Basic K-means objective $\tilde{F}(S, m)$ (3.15, 3.31) for $\{\tilde{\phi}_p\}$ is equivalent to k KM energy $\tilde{F}(S)$ (3.21, 3.35) for kernel $\tilde{\mathcal{K}} \approx \mathcal{K}$ and, therefore, approximates the original kernel clustering objective

$$\tilde{F}(S, \mu_S) \stackrel{c}{=} \tilde{F}(S) \approx F(S) \stackrel{c}{=} E_A(S).$$

Theorem 1 gives unary (linear) bound $\tilde{F}(S, \mu_t)$ (3.46, 3.47) for objective $\tilde{F}(S)$ (3.15, 3.31). We refer to $\tilde{F}(S, \mu_t)$ as a *spectral auxiliary function* for approximate optimization of $E_A(S)$ (last column in Table 3.5). We will also simply call $\tilde{F}(S, \mu_t)$ a *spectral bound* for E_A , not to be confused with a similar term used for matrix eigenvalues.

Theorem 3 (spectral bound for E_A). *For (symmetric) affinity matrix A assume sufficiently large diagonal shift δ generating p.s.d. kernel \mathcal{K} as in Table 3.5. Then, auxiliary function*

$$\tilde{a}_t(S) = \tilde{F}(S, \mu_t) \tag{3.66}$$

using $\tilde{F}(S, m)$ (3.46, 3.47) with embedding $\{\tilde{\phi}_p\} \subset \mathcal{R}^m$ in Tab. 3.5 is a unary (linear) bound for K -means energy $\tilde{F}(S)$ (3.21, 3.35) approximating objective $E_A(S)$ as $m \rightarrow |\Omega|$.

For $m = |\Omega|$ the spectral bounds (Tab. 3.5) are algebraically equivalent to our kernel bounds (Tab. 3.4) since $\tilde{\mathcal{K}} = \mathcal{K}$, see (3.57). Yet, their numerical representation is different. For $m < |\Omega|$ we obtain a range of approximate spectral bounds since $\tilde{\mathcal{K}} \approx \mathcal{K}$ and $\tilde{F} \approx F$. Fig. 3.8 summarizes the relation between our spectral and kernel bounds for E_A . Matlab code for kernel bound and spectral bound is provided in Appendix A.1 and A.2.

Interestingly, Sec. 3.4.3 shows that optimization of our spectral bounds for $m = K$ is algorithmically similar to the common K -means discretization heuristic in spectral relaxation solutions for kernel clustering. Thus, our spectral bound optimization can be seen as a principled formulation justifying this heuristic post-processing step.

Similarly to *kernel bound* in Section 3.3.2, *spectral bound* is useful for optimizing joint energy (3.1). We can iteratively minimize energy $E(S)$ in (3.1) by applying bound optimization approach to its *spectral approximation*

$$\tilde{E}(S) = \tilde{F}(S) + \gamma \sum_{c \in \mathcal{F}} E_c(S_c) \tag{3.67}$$

or its *weighted spectral approximation*

$$\tilde{E}(S) = \tilde{F}^w(S) + \gamma \sum_{c \in \mathcal{F}} E_c(S_c). \tag{3.68}$$

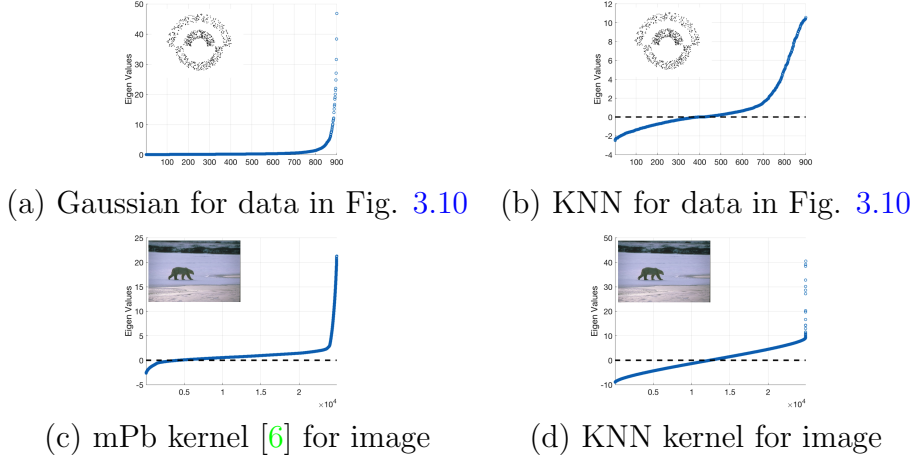


Figure 3.11: Spectrum of eigenvalues of typical kernel matrices for synthetic data (top row) or real image color (bottom row). This helps us to select approximate embedding so as to have small approximation error (3.57). For example, with fixed width gaussian kernel in (a), it suffices to select a few top eigenvectors since the remaining eigenvalues are negligible. Note that the spectrum elevates with increasing diagonal shift δ in (3.60). In principle, we can find the optimal shift for a given number of dimensions m to minimize approximation error.

Corollary 3 (spectral bound for (3.1)). *For any (symmetric) affinity matrix A assume sufficiently large diagonal shift δ generating p.s.d. kernel \mathcal{K} as in Table 3.5. Then, auxiliary function*

$$\tilde{a}_t(S) = \tilde{F}(S, \mu_t) + \gamma \sum_{c \in \mathcal{F}} E_c(S_c) \quad (3.69)$$

using $\tilde{F}(S, m)$ (3.46, 3.47) with embedding $\{\tilde{\phi}_p\} \subset \mathcal{R}^m$ in Tab. 3.5 is a bound for joint energy (3.67, 3.68) approximating (3.1) as $m \rightarrow |\Omega|$.

Approximation quality (3.57) depends on omitted eigenvalues λ_i for $i > m$. Representative examples in Fig. 3.11 show that relatively few eigenvalues may dominate the others. Thus, practically good approximation with small m is possible. Larger m are computationally expensive since more eigenvalues/vectors are needed. Interestingly, smaller m may give better optimization since K-means in higher-dimensional spaces may be more sensitive to local minima. Thus, spectral bound optimization for smaller m may find solutions with lower energy, see Fig. 3.12, even though the quality of approximation is better for larger m .

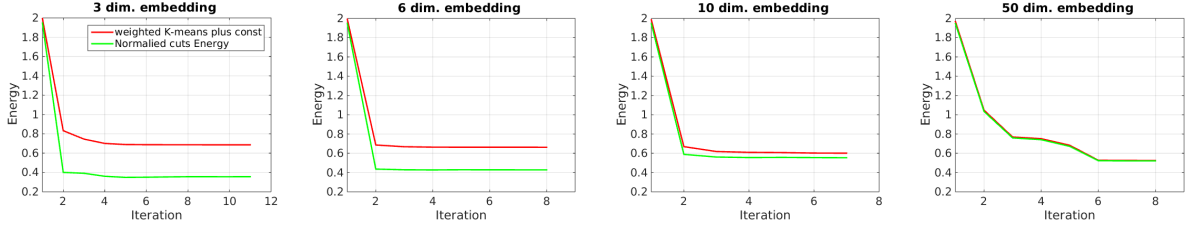


Figure 3.12: For data and affinity matrix in Fig. 3.10, we run weighted K-means with our approximate embedding. The approximation errors $\|\mathcal{K} - \tilde{\mathcal{K}}\|_F^2 / \|\mathcal{K}\|_F^2$ for 3, 6, 10 and 50 dim. embedding are 58%, 41%, 27% and 3% respectively. We compute weighted K-means energy (up to a const) and normalized cuts energy for solution obtained at each iteration. We observed that normalized cuts energy indeed tends to decrease during iterations of K-means. Even 10 dim. embedding gives good alignment between K-means energy and normalized cuts energy. Higher dimensional embedding gives better energy approximation, but not necessarily better solution with lower energy.

Similarly to the kernel bound algorithms discussed in Section 3.3.3 one can optimize the approximate spectral bound (3.69) for energy (3.1) using standard algorithms for regularization. This follows from the fact that the first term in (3.69) is unary (linear). Algorithm 4 shows a representative (approximate) bound optimization technique for (3.1) using move-making algorithms [31]. Note that for $\gamma = 0$ (no regularization terms) our bound optimization Algorithm 4 reduces to basic K-means over approximate isometry embeddings $\{\tilde{\phi}_p\} \subset \mathcal{R}^m$ similar but not identical to common discretization heuristics in spectral relaxation methods.

Some extensions for optimization ideas in Sec. 3.3 and 3.4 are discussed in [221]. For example, diagonal shift δ can be used to reduce Frobenius error (3.57). We also discuss *pseudo-bounds* [217].

3.4.3 Relation to spectral clustering

Our approximation of kernel clustering such as NC via basic KM over low dimensional embeddings $\tilde{\phi}_p$ is closely related to popular spectral clustering algorithms [211, 171, 14] using eigen decomposition for various combinations of kernel, affinity, distortion, laplacian, or other matrices. Other methods also build low-dimensional Euclidean embeddings [171, 14, 253] for basic KM using motivation different from isometry and approximation errors with respect to given affinities. We are mainly interested in discussing relations to spectral

Algorithm 4: α -Expansion for Spectral Cut

Input : Affinity matrix \mathcal{A} of size $|\Omega| \times |\Omega|$;
Initial labeling S_0^1, \dots, S_0^K

Output: S^1, \dots, S^K : partition of the set Ω

- 1 Find top m eigenvalues/vectors Λ^m, V^m for a matrix in the 4th col. of Tab. 3.5 ;
 - 2 Compute embedding $\{\tilde{\phi}_p\} \subset \mathcal{R}^m$ for some δ and set $t := 0$;
 - 3 **while** *not converged* **do**
 - 4 Set $\tilde{a}_t(S)$ to be spectral bound (3.69) at current partition S_t ;
 - 5 **for** *each label* $\alpha \in \mathcal{L} = \{1, \dots, K\}$ **do**
 - 6 Find $S_t := \arg \min \tilde{a}_t(S)$ within one α expansion of S_t ;
 - 7 **end**
 - 8 Set $t := t + 1$;
 - 9 **end**
-

methods approximately optimizing kernel clustering criteria such as AA, AC, and NC [211].

Many spectral relaxation methods also use various eigen decompositions to build explicit data embeddings followed by basic K-means. In particular, the smallest or largest eigenvectors for the (generalized) eigenvalue problems in Table 3.6 give well-known exact solutions for the relaxed problems. In contrast to our approach, however, the final K-means stage in spectral methods is often presented without justification [211, 235, 6] as a heuristic for quantizing the relaxed continuous solutions into a discrete labeling. It is commonly understood that

“... there is nothing principled about using the K-means algorithm in this step”
(Sec. 8.4 in [235])

or that

“... K-means introduces additional unwarranted assumptions.” (Sec. 4 in [251])

Also, typical spectral methods use K eigenvectors solving the relaxed K -cluster problems followed by KM quantization. In contrast, we choose the number of eigenvectors m based on Frobenius error for isometry approximation (3.57). Thus, the number m is independent from the predefined number of clusters.

Below we juxtapose our approximate isometry low dimensional embeddings in Table 3.5 with embeddings used for ad-hoc discretization by the standard spectral relaxation methods in Table 3.6. While such embeddings are similar, they are not identical. Thus, our

Frobenius error argument offers a justification and minor corrections for KM heuristics in spectral methods, even though the corresponding methodologies are unrelated. More importantly, our bound formulation allows integration of kernel clustering with additional regularization constraints (3.1).

Embeddings in spectral methods for NC: Despite similarity, there are differences between our low-dimensional embedding (3.65) provably approximating kernel $\mathcal{K} = \delta D^{-1} + D^{-1}AD^{-1}$ for the k KM formulation of NC [9, 64] and common ad-hoc embeddings used for KM discretization step in the spectral relaxation methods. For example, one such discretization heuristic [211, 235] uses embedding $\tilde{\phi}_p$ (right column in Tab. 3.6) defined by the columns of matrix U^K whose rows are the K top (unit) eigenvectors of the standard eigen system (left column). It is easy to verify that the rows of matrix $VD^{-\frac{1}{2}}$ are non-unit eigenvectors for the generalized eigen system for NC. The following relationship

$$\tilde{\phi}_p = U^K \equiv [V^K D^{-\frac{1}{2}}]^{rn}$$

where operator $[\cdot]^{rn}$ normalizes matrix rows, demonstrates certain differences between ad hoc embeddings used by many spectral relaxation methods in their heuristic K-means discretization step and justified approximation embedding (3.65) in Tab. 3.5. Note that our formulation scales each embedding dimension, *i.e.* rows in matrix $V^K D^{-\frac{1}{2}}$, according to eigenvalues instead of normalizing these rows to unit length.

There are other common variants of embeddings for the K-means discretization step in spectral relaxation approaches to the normalized cut. For example, [16, 155, 6] use

$$\tilde{\phi}_p = [\Lambda^{-\frac{1}{2}}U]_p^K$$

for discretization of the relaxed NC solution. The motivation comes from the physics-based *mass-spring system* interpretation [16] of the generalized eigenvalue system.

Some spectral relaxation methods motivate their discretization procedure differently. For example, [251, 9] find the closest integer solution to a *subspace* of equivalent solutions for their particular very similar relaxations of NC based on the same eigen decomposition (3.63) that we used above. Yu and Shi [251] represent the subspace via matrix

$$X' \equiv [\sqrt{\Lambda^m} V^m D^{-\frac{1}{2}}]^{cn}$$

where columns differ from our embedding $\tilde{\phi}(I_p)$ in (3.65) only by normalization. Theorem 1 by Bach and Jordan [9] equivalently reformulates the distance between the subspace and integer labelings via a *weighted* K-means objective for embedding

$$\tilde{\phi}_p = \sqrt{\frac{1}{d_p}} V_p^m \quad (3.70)$$

	spectral relaxation [211]	common discretization heuristic [235] (embedding & K-means)		
AA	$A\mathbf{u} = \lambda\mathbf{u}$	$\tilde{\phi}_p := U_p^K$	$\equiv V_p^K$	$\Leftarrow V'\Lambda V = A$
AC	$(D - A)\mathbf{u} = \lambda\mathbf{u}$	$\tilde{\phi}_p := U_p^K$	$\equiv V_p^K$	$\Leftarrow V'\Lambda V = D - A$
NC	$(D - A)\mathbf{u} = \lambda D\mathbf{u}$	$\tilde{\phi}_p := U_p^K$	$\equiv [V^K D^{-\frac{1}{2}}]_p^{rn}$	$\Leftarrow V'\Lambda V = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$

Table 3.6: *Spectral relaxation* and discretization heuristics for objectives for kernel clustering objectives $E_A(S)$ for affinity A . The corresponding *degree* matrix D is diagonal with elements $d_p := \sum_q A_{pq}$. To extract integer labeling from the relaxed solutions produced by the eigen systems (second column), spectral methods often apply basic KM to some ad hoc data embedding $\tilde{\phi}$ (last column) based on the first K unit eigenvectors \mathbf{u} , the rows of matrix U^K . While our main text discusses some variants, the most basic idea [211, 235] is to use the columns of U^K as embedding $\tilde{\phi}_p$. For easier comparison, the last column also shows equivalent representations of this embedding based on the same eigen decompositions $V'\Lambda V$ as those used for our isometry eigenmaps in Tab. 3.5. In contrast, our embeddings are derived from justified approximations of the original non-relaxed AA, AC, or NC objectives. Note that NC corresponds to a *weighted* case of K-means with data point weights $w_p = d_p$ [9, 64], see (3.41) in Sec. 3.2.3.

and weights $w_p = d_p$. This embedding is different from (3.65) only by eigenvalue scaling.

Interestingly, a footnote in [9] states that NC objective (3.40) is equivalent to weighted KM objective (3.31) for exact isometry embedding

$$\phi_p = \frac{1}{d_p} G_p \quad \in \mathcal{R}^{|\Omega|} \quad (3.71)$$

based on any decomposition $A \equiv G'G$. For example, our exact isometry map (3.65) for $m = |\Omega|$ and $G = \sqrt{\Lambda} V D^{\frac{1}{2}}$ is a special case. While [9] reduce NC to K-means¹¹, their low-dimensional embedding $\tilde{\phi}$ (3.70) is derived to approximate the subspace of relaxed NC solutions. In contrast, low-dimensional embedding (3.65) approximates the exact isometry map ϕ ignoring relaxed solutions. It is not obvious if decomposition $A \equiv G'G$ for the exact embedding (3.71) can be used to find any approximate lower-dimensional embeddings like (3.65).

¹¹ KM procedure (3.22) (weighted version) is not practical for objective (3.31) for points ϕ_p in $\mathcal{R}^{|\Omega|}$. Instead, Dhillon et al. [64] later suggested *pairwise* KM procedure (3.23) (weighted version) using kernel $\mathcal{K}_{pq} \equiv \langle \phi_p, \phi_q \rangle$.

3.5 Experiments

This section is divided into two parts. The first part (Sec.3.5.1) shows the benefits of extra MRF regularization for kernel & spectral clustering, e.g. normalized cut. We consider pairwise Potts, label cost and robust bin consistency term, as discussed in Sec.3.2.1. We compare to spectral clustering [211, 155] and kernel K-means [64], which can be seen as degenerated versions for spectral and kernel cuts (respectively) without MRF terms. We show that MRF helps kernel & spectral clustering in segmentation and image clustering. In the second part (Sec.3.5.2) we replace the log-likelihoods in model-fitting methods, *e.g.* GrabCut [200], by kernel clustering term, *e.g.* AA and NC. This is particularly advantageous for high dimension features (location, depth, motion).

Implementation details: For segmentation, our kernel cut method uses either Gaussian kernels of fixed bandwidth σ or common KNN kernels with adaptive bandwidth *e.g.* see [256, 22] and [158]. Pixel features I_p can be concatenation of LAB (color), XY (location) and M (motion or optical flow) [33]. For KNN, we choose 400 neighbors for each pixels and randomly sample 50 neighbors for efficiency. Sampling does not degrade our segmentation but expedites bound evaluation. We also use popular *mPb* contour based affinities [6]. The window radius is set to 5 pixels.

Another detail to mention is diagonal shift of the kernel matrix. It is necessary to give PSD matrix so that our bounds hold. However, in practice, we find the energies to decrease at each iteration even without any diagonal shift for some kernels that are not necessarily PSD, *e.g.* KNN kernel. As such, we choose not to add any diagonal shift in our experiments bellow. Also adding too large a diagonal shift may lead to poor local minima in kernel K-means algorithm, as discussed in [65].

For regularization in (3.2) we use standard contrast-sensitive penalty $w_{pq} = \frac{1}{d_{pq}} e^{-0.5\|I_p - I_q\|_2^2/\eta}$ [30] where η is the average of $\|I_p - I_q\|_2^2$ over a 8-connected neighborhood and d_{pq} is the distance between pixels p and q in the image plane. We set $w_{pq} = \frac{1}{d_{pq}}$ for length regularization.

We compare kernel clustering term $E_A(S)$ in (3.1) with a standard model-fitting term (3.5) using histogram-based probability model, as is common in Grabcut approach [234, 140]. We tried various bin size for spatial and depth channels.

With fixed width Gaussian kernel, the time complexity of the naive implementation of kernel bound evaluation in (3.54) is $O(|\Omega|^2)$. The bottleneck is the evaluation of $\mathcal{K}X_t$ and $X_t'\mathcal{K}X_t$ in derivative $\nabla\hat{e}(X_t)$ (3.49). In this case, we resort to fast approximate dense filtering method in [181], which takes $O(|\Omega|)$ time. Also notice that the time complexity of the approach in [181] grows exponentially with data dimension N . A better approach

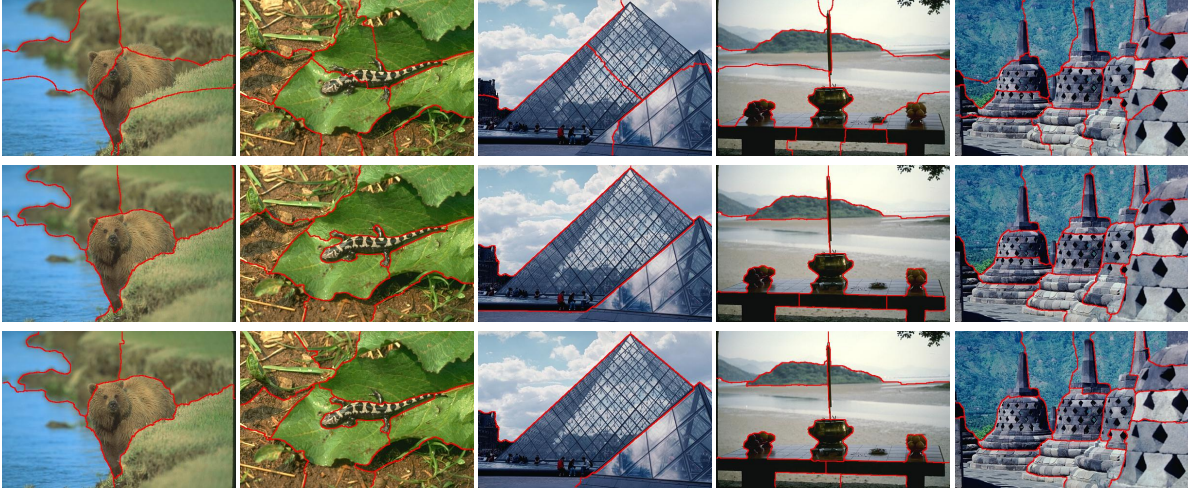


Figure 3.13: Sample results on BSDS500. Top row: spectral clustering. Middle & Bottom rows: our Kernel & Spectral Cuts.

for high-dimensional dense filtering is proposed in [3], which is of time $O(|\Omega| \times N)$. We use [181] for low-dimensional color spaces.

3.5.1 MRF helps Kernel & Spectral Clustering

Here we add MRF regulation terms to typical normalized cut applications, such as unsupervised multi-label segmentation [6] and image clustering [51]. Our kernel and spectral cuts are used to optimize the joint energy of normalized cut and MRF (3.1) or (3.68).

Normalized Cut with Potts Regularization

Spectral clustering [211] typically solves a (generalized) eigen problem, followed by simple clustering method such as K-means on the eigenvectors. However, it is known that such paradigm results in undesirable segmentation in large uniform regions [6, 155], see examples in Fig. 3.13. Obviously such edge mis-alignment can be penalized by contrast-sensitive Potts term. Our spectral and kernel cuts get better segmentation boundaries. As is in [64] we use spectral initialization.

Tab. 3.7 gives quantitative results on BSDS500 datasal. Number of ground truth segments is provided to each method. Kernel and spectral cuts give better covering, PRI

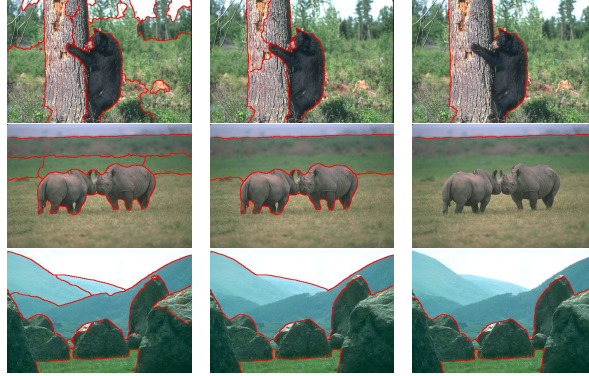


Figure 3.14: Segmentation using our kernel cut with label cost. We experiment with increasing value of label cost h_k for each label (from left to right)

(probabilistic rand index) and VOI (variation of information) than spectral clustering. Fig. 3.13 gives sample results. Kernel K-means [64] gives results similar to spectral clustering and hence are not shown.

Normalized Cuts with Label Cost [60]

Unlike spectral clustering, our kernel and spectral cuts do not need the number of segments beforehand. We use kernel cut to optimize a combination of the normalized cut, Potts model and label costs terms. The label cost (3.4) penalizes each label by constant h_k . The energy is minimized by α -expansion and $\alpha\beta$ -swap moves in Sec.3.3.3. We sample initial models from patches, as in [60]. Results with different label cost are shown in Fig. 3.14. Due to sparsity prior, our kernel and spectral cuts automatically prune *weak* models and determine the number of segments, yet yield regularized segmentation. We use *KNN* affinity for normalized cut and mPb [6] based Potts regularization.

method	Covering	PRI	VOI
Spectral Clustering	0.34	0.76	2.76
Our Kernel Cut	0.41	0.78	2.44
Our Spectral Cut	0.42	0.78	2.34

Table 3.7: Results of spectral clustering (K-means on eigenvectors) and our Kernel Cut & Spectral Cuts on BSDS500 dataset. For this experiment mPb-based kernel is used [6].

Normalized Cut with High-Order Consistency

It is common that images come with multiple tags, such as those in Flickr platform or the LabelMe dataset [178]. We study how to utilize tag-based group prior for image clustering [51] enforced as a high-order consistency potential common in MRF-based image segmentation [120, 183, 220].

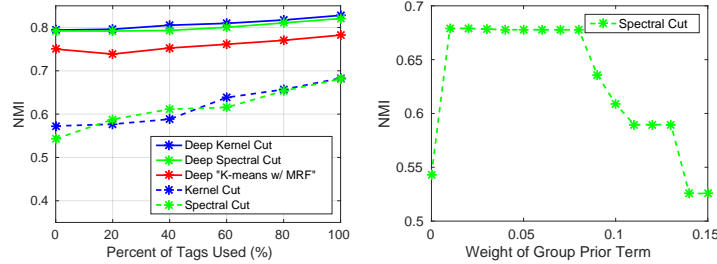


Figure 3.15: Incorporating group prior achieves better NMI for image clustering. Here we use tags-based group prior. Our method achieved better NMI when more images are tagged. The right plot shows how the weight of bin consistency term affects our method.

We experiment on the LabelMe dataset [178] which contains 2,600 images of 8 scene categories (coast, mountain, forest, open country, street, inside city, tall buildings and highways). We use the same GIST feature, affinity matrix and group prior as used in [51]. We found the group prior to be noisy. The dominant category in each group occupies only 60%-90% of the group. The high-order consistency term is defined on each group. For each group, we introduce an energy term that is akin to the *robust* P^n -Potts [120], which can be exactly minimized within a single $\alpha\beta$ -swap or α -expansion move. Notice that here we have to use robust consistency potential instead of rigid ones.

Our kernel cut minimizes NC plus the *robust* P^n -Potts term. Spectral cut minimizes energy of (3.67). Normalized mutual information (NMI) is used as the measure of clustering quality. Perfect clustering with respect to ground truth has NMI value of 1.

Spectral clustering and kernel K-means [64] give NMI value of 0.542 and 0.572 respectively. Our kernel cut and spectral cut significantly boost the NMI to 0.683 and 0.681. Fig. 3.15 shows the results with respect to different amount of image tags used. The left most points correspond to the case when no group prior is given. We optimize over the weight of high order consistency term, see Fig. 3.15. Note that it's not the case the larger the weight the better since the grouping prior is noisy.

We also utilize deep features, which are 4096 dimensional fc7 layer from AlexNet [131].

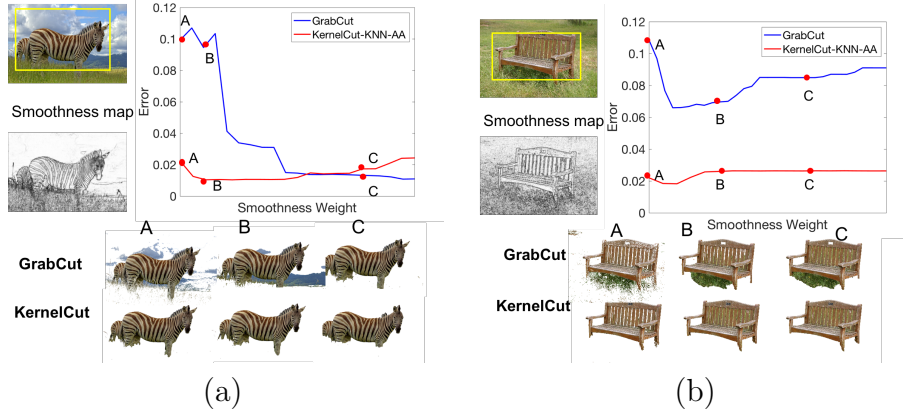


Figure 3.16: Illustration of robustness to smoothness weight.

We either run plain K-means, or construct a *KNN* kernel on deep features. These algorithms are denoted as deep K-means, deep spectral cut or deep kernel cut in Fig. 3.15. Incorporating group prior indeed improved clustering. The best NMI of 0.83 is achieved by our kernel cut and spectral cut for KNN kernel on deep features.

3.5.2 Kernel & Spectral Clustering helps MRF

In typical MRF applications we replace the log-likelihood terms by average association or normalized cut. We evaluate our Kernel Cut (fixed width kernel or *KNN*) in the context of interactive segmentation, and compare with the commonly used GrabCut algorithm [200]. In Sec. 3.5.2, we show that our kernel cut is less sensitive to choice of regularization weight γ . We further report results on the GrabCut dataset of 50 images and the Berkeley dataset in Sec. 3.5.2. We experiment with both (i) contrast-sensitive edge regularization, (ii) length regularization and (iii) color clustering (i.e., no regularization) so as to assess to what extent the algorithms benefit from regularization.

From Sec. 3.5.2 to Sec. 3.5.2, we also report segmentation results of our kernel cut with high-dimensional features I_p , including location, texture, depth, and motion respectively.

Robustness to regularization weight

We first run all algorithms without smoothness. Then, we experiment with several values of γ for the contrast-sensitive edge term. In the experiments of Fig. 3.16 (a) and (b), we

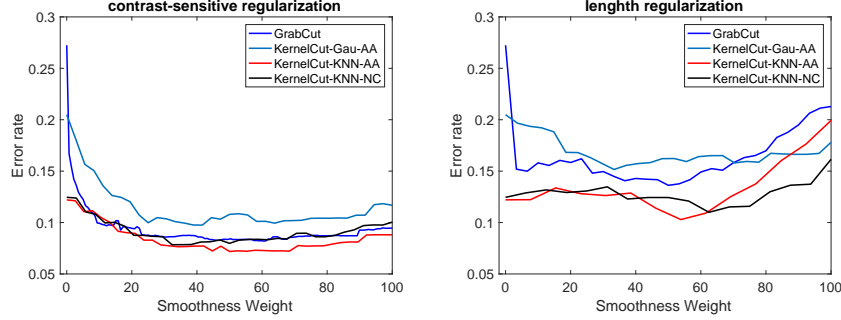


Figure 3.17: Average error vs. regularization weights for different variants of our KernelCut on the GrabCut dataset.

used the yellow boxes as initialization. For a clear interpretation of the results, we did not use any additional hard constraint. In Fig. 3.16, "KernelCut-KNN-AA" means Kernel Cut with KNN kernel for average association (AA). Without smoothness, our Kernel Cut yields much better results than Grab Cut. Regularization significantly benefited the latter, as the decreasing blue curve in (a) indicates. For instance, in the case of the zebra image, model fitting yields a plausible segmentation when assisted with a strong regularization. However, in the presence of noisy edges and clutter, as is the case of the chair image in (b), regularization does not help as much. Note that for small regularization weights γ our method is substantially better than model fitting. Also, our method is less dependent on regularization weight and does not require fine tuning of γ .

Segmentation on GrabCut & Berkeley datasets.

First, we report results on the GrabCut database (50 images) using the bounding boxes provided in [141]. For each image the error is the percentage of mis-labeled pixels. We compute the average error over the dataset.

We experiment with four variants of our Kernel Cut, depending on whether to use fixed width Gaussian kernel or KNN kernel, and also the choice of normalized cut or average association term. We test different smoothness weights and plot the error curves¹² in Fig. 3.17. Tab. 3.8 reports the best error for each method. For contrast-sensitive regularization GrabCut gets good results (8.2%). However, without edges (Euclidean or no regularization) GrabCut gives much higher errors (13.6% and 27.2%). In contrast, KernelCut-KNN-AA

¹²The smoothness weights for different energies are not directly comparable; Fig. 3.17 shows all the curves for better visualization.

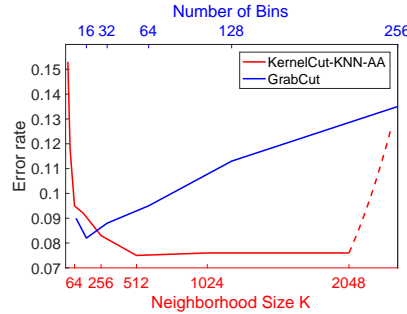


Figure 3.18: Our method aKKM is robust to choice of K while GrabCut is sensitive to bin size for histograms.

boundary smoothness	color clustering term			
	GrabCut	KernelCut -Gau-AA	KernelCut -Gau-NC	KernelCut -KNN-AA
none	27.2	20.4	17.6	12.2
Euclidean length	13.6	15.1	16.0	10.2
contrast-sensitive	8.2	9.7	13.8	7.1

Table 3.8: Box-based interactive segmentation (Fig. 3.19). Error rates (%) are averaged over 50 images in GrabCut dataset. KernelCut-Gau-NC means KernelCut for fixed width Gaussian kernel based normalized cut objective.

(Kernel Cut with adaptive KNN kernel for AA) gets only 12.2% doing a better job in color clustering without any help from the edges. In case of contrast-sensitive regularization, our method outperformed GrabCut (7.1% vs. 8.2%) but both methods benefit from strong edges in the GrabCut dataset. Fig. 3.18 shows that our Kernel Cut is also robust to the hyper-parameter, i.e. K for nearest neighbours, unlike GrabCut.

Fig. 3.19 gives some results. The top row shows a failure case for GrabCut where the solution aligns with strong edges. The second row shows a challenging image where our KernelCut-KNN-AA works well. The third and fourth rows show failure cases for Kernel Cut with fixed-width Gaussian kernel due to Breiman’s bias [158] separating uniform color segments; see green bush and black suit. Adaptive kernel (KNN) addresses this bias.

We also tested seeds-based segmentation on a different database [160] with ground truth, see Tab. 3.9 and Fig. 3.20.



Figure 3.19: Sample results for GrabCut and our kernel cut with fixed width Gaussian or adaptive width KNN kernel, see Tab. 3.8.

boundary smoothness	color clustering term		
	BJ	GrabCut	KernelCut -KNN-AA
none	12.4	12.4	7.6
contrast-sensitive	3.2	3.7	2.8

Table 3.9: Seeds-based interactive segmentation (Fig. 3.20). Error rates (%) are averaged over 82 images from Berkeley database. Methods get the same seeds entered by four users. We removed 18 images with multiple nearly-identical objects from 100 image subset in [160]. (GrabCut and KernelCut-KNN-AA give 3.8 and 3.0 errors on the whole database.)

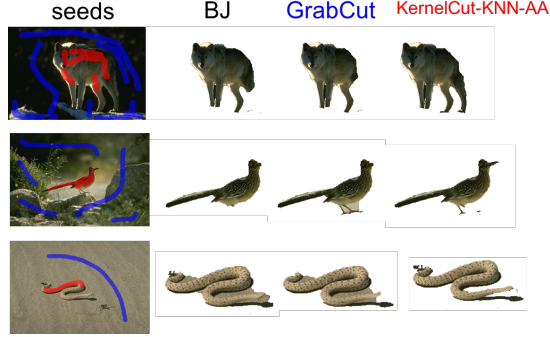


Figure 3.20: Sample results for BJ [30], GrabCut [200], and our kernel cut for adaptive KNN kernel, see Tab. 3.9.



Figure 3.21: Visualization of a pixel's K-Nearest-Neighbours for RGB feature (left) or RGBXY feature (right).

Segmentation of similar appearance objects

Even though objects may have similar appearances or look similar to the background (*e.g.* the top row in Fig. 3.23), we assume that the objects of interest are compact and have different locations. This assumption motivates using XY coordinates of pixels as extra features for distinguishing similar or camouflaged objects. XY features have also been used in [174] to build space-variant color distribution. However, such distribution used in MRF-MAP inference [174] would still over-fit the data [218]. Let $I_p \in \mathcal{R}^5$ be the augmented color-location features $I_p = [l_p, a_p, b_p, \beta x_p, \beta y_p]$ at pixel p where $[l_p, a_p, b_p]$ is its color, $[x_p, y_p]$ are its image coordinates, and β is a scaling parameter. Note that the edge-based Potts model [30] also uses the XY information. Location features in the clustering and regularization terms have complementary effect: the former solves appearance camouflage while the latter gets edge alignment.

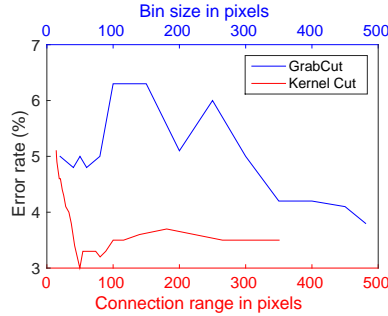


Figure 3.22: Error on Multi-objects dataset. We vary spatial bin-size for GrabCut and weight β in $[l, a, b, \beta X, \beta Y]$ for Kernel Cut. The connection range is the average geometric distance between a pixel and its k^{th} nearest neighbor. **The right-most point of the curves corresponds to the absence of XY features.** GrabCut does not benefit from XY features. Kernel Cut achieves the best error rate of 2.9% with connection range of 50 pixels.

We test the effect of adding XY into feature space for GrabCut and Kernel Cut. We try various β for Kernel Cut. Fig. 3.21 shows the effect of different β on $KNNs$ of a pixel. For histogram-based GrabCut we change spatial bin size for the XY channel, ranging from 30 pixels to the image size. We report quantitative results on 18 images with similar objects and camouflage from the Berkeley database [159]. Seeds are used here. Fig. 3.22 shows average errors for multi-object dataset. Fig. 3.23 gives multi-label segmentation of similar objects in one image with seeds using our algorithm. We optimize kernel bound with move-making for NC and smoothness term combination, as discussed in Sec. 3.3.2. Fig. 3.23 (c) shows energy convergence.

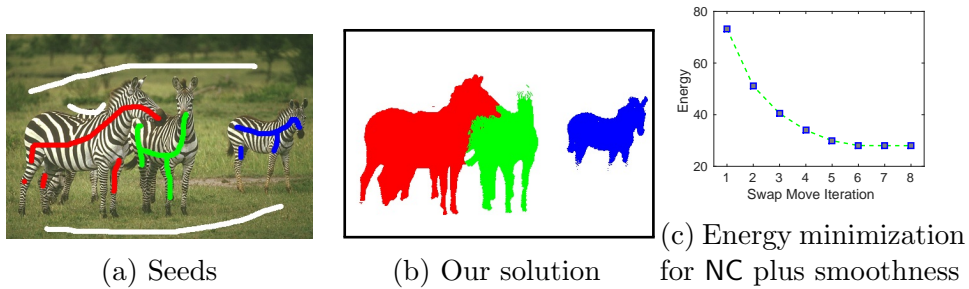


Figure 3.23: Multi-label segmentation for similar objects.

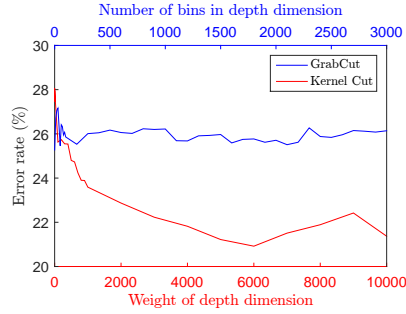


Figure 3.24: The average errors of GrabCut and Kernel Cut methods over 64 images selected from NYUv2 database [169].

Interactive RGBD Images Segmentation

Depth sensor are widely used in vision for 3D modelling [67, 170], semantic segmentation [62, 100, 169, 194], motion flow [94]. We selected 64 indoor RGBD images from semantic segmentation database NYUv2 [169] and provided bounding boxes and ground

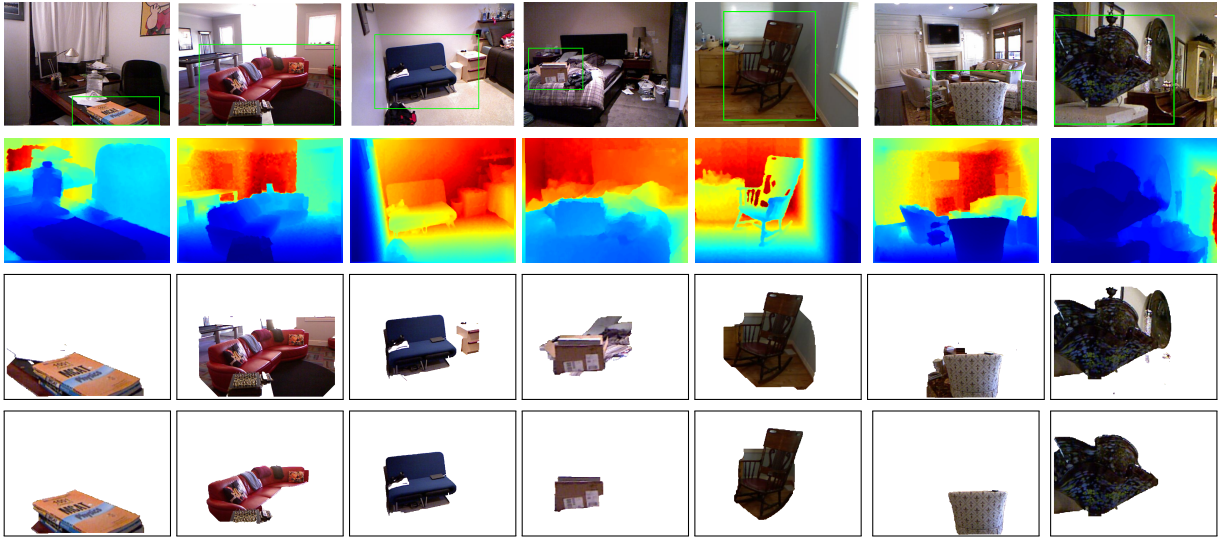


Figure 3.25: RGBD+XY examples. The first two rows show original images wit bounding box and color-coded depth channel. The third row shows the results of Grabcut, the forth row shows the results of Kernel Cut. The parameters of the methods were independently selected to minimize their average error rates over the database.

truth. In contrast to [200], the prepared dataset consists of low-quality images: there are camera motion artifacts, underexposed and overexposed regions. Such artifacts make color-based segmentation harder.

We compare GrabCut to Kernel Cut over joint features $I_p = [L_p, a_p, b_p, \beta D_p]$ as in Sec.3.5.2. Figs. 3.24 and 3.25 show the error statistics and segmentation examples. While Kernel Cut takes advantage of the additional channel, GrabCut fails to improve.

Motion segmentation

Besides the location and depth features, we also test segmentation with motion features. Figs. 3.26, 3.27 and 3.28 compare motion segmentations using different feature spaces: RGB, XY, M (optical flow) and their combinations (RGBM or RGBXY or RGBXYM).

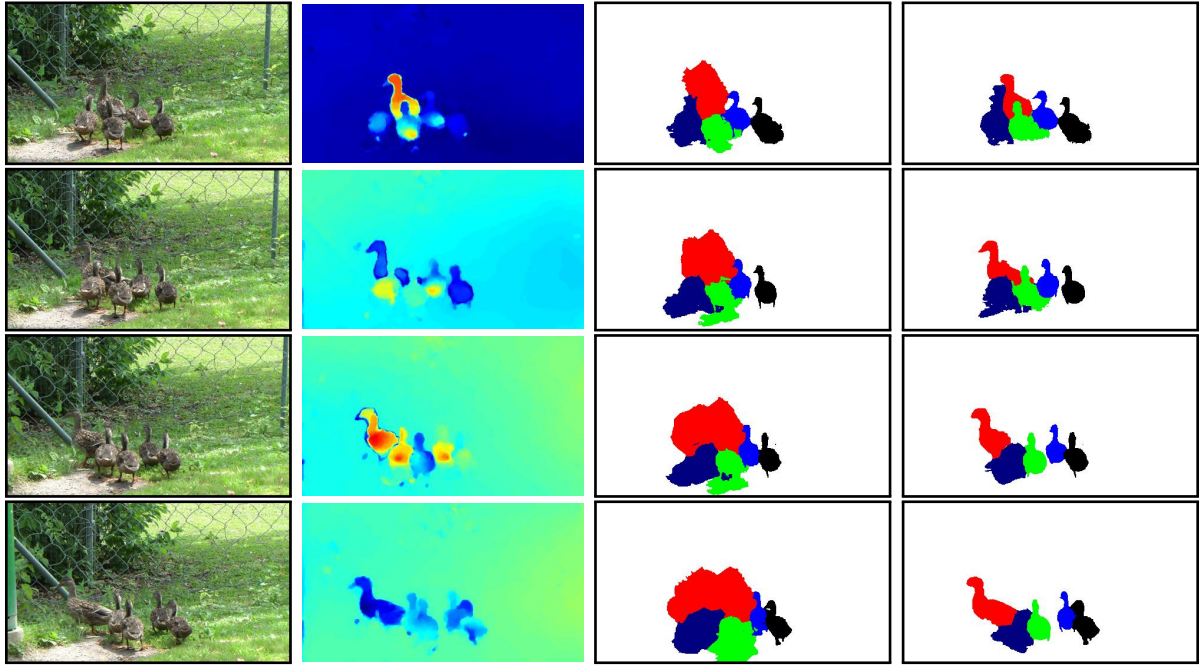


(a) frames (b) optical flow [33] (c) **M+XY** (d) **RGB+XY** (e) **RGBM+XY**

Figure 3.26: Motion segmentation using our framework for the sequence *horses01* in FBMS-59 dataset [34]. Motion feature alone (**M+XY** in (c)) is not sufficient to obtain fine segmentation. Our framework successfully utilize motion feature (optical flow) to separate the horse from the barn, which have similar appearances. See supplementary material for results on the video.

Abbreviation $+XY$ means Potts regularization. We apply kernel cut (Alg.3) to the combination of NC with the Potts term.

Challenging video examples: For videos in FBMS-59 dataset [34], our algorithm runs on individual frames instead of 3D volume. Segmentation of previous frame initializes the next frame. The strokes are provided *only* for the first frame. We use the optical flow algorithm in [33] to generate M features. Selected frames are shown in Figs. 3.26 and 3.27. Instead of tracks from all frames in [176], our segmentation of each frame uses only motion estimation between two consecutive frames. Our approach jointly optimizes normalized cut and Potts model. In contrast, [176] first clusters semi-dense tracks via spectral clustering [34] and then obtains dense segmentation via regularization.



(a) frames (b) optical flow [33] (c) $\mathbf{RGBXY}+XY$ (d) $\mathbf{RGBXYM}+XY$

Figure 3.27: Multi-label motion segmentation using our framework for the sequence *ducks01* in FBMS-59 dataset [34]. This video is challenging since the ducks have similar appearances and even spatially overlap with each other. However, different ducks come with different motions, which helps our framework to better separate individual ducks. See supplementary materials.

Kitti segmentation example: We also experiment with Kitti dataset [161]. Fig. 3.28 shows the multi-label segmentation using either color information RGB+XY (first row) or motion MXY+XY (second row). The ground-truth motion field works as M channel. Note that the motion field is known only for approximately 20% of the pixels. To build an affinity graph, we construct a *KNN* graph from pixels that have motion information. The regularization over 8-neighborhood on the pixel grid interpolates the segmentation labels during the optimization procedure.

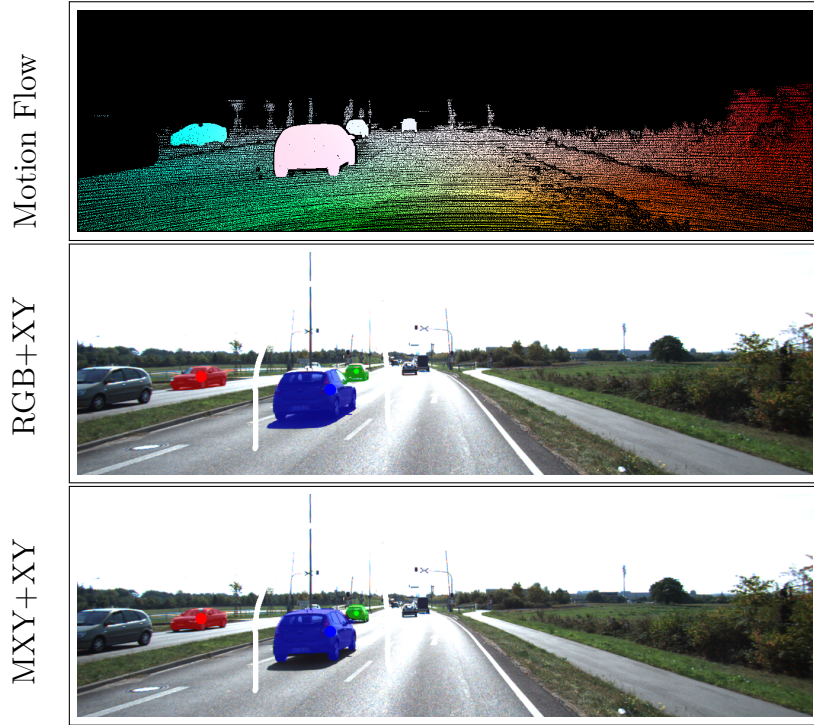


Figure 3.28: Motion segmentation for image 000079_10 from *KITTI* [161] dataset. The first row shows the motion flow. Black color codes the pixels that do not have motion information. The second row shows color-based segmentation. The third row shows motion based segmentation with location features. We also tried M+XY segmentation, but it does not work as well as MXY+XY above. The results for RGBMXY+XY were not significantly different from MXY+XY.

Chapter 4

Regularized Losses for Weakly Supervised CNN Segmentation

Most recent semantic segmentation methods train deep convolutional neural networks with tens of thousands of fully annotated masks, which is laborious to obtain. Weakly-supervised CNN segmentation given e.g. scribbles or image-level tags has increasingly attracted researchers’ attention. Common approaches mimic full supervision via “fake” fully-labeled masks (proposals) generated from available partial input. To obtain such full masks, the typical methods explicitly use standard regularization techniques in “shallow” segmentation, e.g. graph cut or mean-field for CRF inference. In contrast, we integrate such standard segmentation regularizers and criterion directly into the loss, for example pair-wise MRF energy and normalized cut clustering criterion. Minimization of regularized losses is a principled approach to weak supervision well-established in deep learning, in general. We propose and experimentally compare different regularized losses. Besides, we propose alternative direction method (ADM) as better optimization beyond gradient descent (GD) for certain type of regularized losses, e.g. Grid CRF, for which GD is not an effective optimizer. Our regularized loss approach achieves state-of-the-art accuracy in semantic segmentation with near full-supervision quality.

4.1 Introduction and Motivation

Since the seminal work [131], deep convolutional neural networks (CNN) dominate almost all aspects of computer vision, e.g. recognition [216, 102], detection [84, 193], and segmen-

tation [150, 45]. It is capable of learning intermediate representations at different levels given abundant training data. Typically, pixel-wise cross entropy loss (1.13) is minimized.

Supervised training of FCNs requires a huge number of fully annotated ground-truth masks that is costly to obtain. Training with weak annotations, e.g. scribbles [147, 244], bounding boxes [116, 58, 244, 180], clicks [13], and image-level tags [244, 180], has caught a lot of interest recently.

Weakly supervised semantic segmentation is commonly addressed by mimicking full supervision via synthesizing fully-labeled training masks (proposals) from the available partial inputs [192, 180, 147]. These schemes typically iterate between two steps: CNN training and proposal generation via regularization-based shallow interactive segmentation, e.g. graph cut [30, 200, 147, 116] for grid CRF or mean-field inference [192, 180, 130, 121] for dense CRF. However, inaccuracies of such masks (segmentation proposals) mislead training since typical cross entropy (CE) loss is minimized over mislabeled points and the network over-fits mistakes, see Fig. 4.3 in Sec.4.2.2.

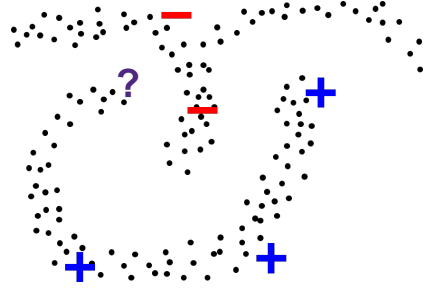
In contrast, we integrate shallow regularizers directly into the loss for CNN segmentation. The regularization loss on unlabeled pixels is combined with partial cross entropy loss on labeled pixels to give a *regularized loss*. The use of unsupervised loss terms acting as regularizers on the output of deep-learning architectures is a principled approach to exploit structure similarity of partially labeled data [241, 89]. Surprisingly, this general idea was largely overlooked in weakly-supervised CNN segmentation where current methods often introduce computationally expensive CRF pre-processing [116] or layers [121] generating “fake” full masks from partial input.

Our regularized loss approach is simple, efficient, and generally applicable to a wide range of segmentation regularization as we demonstrated through experiments. We propose different regularization losses including MRF energy, clustering criterion, and volume constraint. Such losses can be adapted to many forms of weak (or semi) supervision. Here, we mostly focus on training with partial masks (user scribble) where regularization losses combined with cross entropy over the partial masks achieve the state-of-the-art close to full-supervision quality.

It is straightforward to minimize regularized loss via gradient descent in CNN training. However, it is well-known that gradient descent leads to poor local minima for many regularizers, e.g. Grid CRF, in shallow segmentation and many stronger algorithms were proposed [30, 31, 124, 113]. Similarly, we show better optimization beyond GD. We propose alternative direction method (ADM) for optimizing regularized losses. Interestingly, our ADM optimization described in Sec. 4.4 makes connections between proposal-generation heuristics and our regularized losses framework.



(a) Weakly-supervised Segmentation



(b) Semi-supervised Learning

Figure 4.1: We approach the problem of weakly-supervised segmentation (a) with scribbles through principled techniques in semi-supervised learning (b).

Our regularized loss approach to weakly-supervised CNN segmentation is motivated both by common regularization for weakly-supervised segmentation in Sec. 4.1.1 and also regularization for semi-supervised learning in Sec. 4.1.2. The problems of weakly-supervised segmentation and semi-supervised learning are highly related, see Fig. 4.1.

4.1.1 Regularization for Weakly-supervised Segmentation

Due to the importance of low-level regularizers for segmentation and computer vision in general, there are many choices available in the literature [82, 42, 30, 211, 97, 163, 189, 132, 48, 222]. To further motivate our approach, we discuss one specific and basic example of such regularizer.

Probably the simplest MRF regularization for segmentation is the pairwise regularization [30] which can be combined with constraints over seeds $p \in \Omega_{\mathcal{L}}$ of its predefined labels in the form of indicator vector $Y_p \in \{0, 1\}^K$,

$$\min_S \sum_{p \in \Omega_{\mathcal{L}}} \sum_k -S_p^k \cdot \log Y_p^k + \lambda \sum_{p, q \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q] \quad (4.1)$$

where $S_p^k \in \{0, 1\}$ are now interpreted as class assignment indicators, $[\cdot]$ are Iverson brackets, and \mathcal{N} is a neighborhood system. The first term enforces S_p to agree with Y_p for $p \in \Omega_{\mathcal{L}}$ otherwise the cost is infinite ($-\log 0$). The regularization term penalizes disagreements between pairs of points and the weight w_{pq} is contrast-sensitive. It prefers segmentation that align with object boundaries and edges. There are many possible solutions that satisfy the hard constraints w.r.t. the seeds, and such ambiguity is eliminated by the regularization term. An example of seeds $\Omega_{\mathcal{L}} \subset \Omega$ is in Fig. 4.1 (a).

The neighborhood system \mathcal{N} is typically defined on the image grid. For sparse non-negative weights w_{pq} for nearest neighbors p and q as in classical Ising or Potts models, there is a geometric interpretation corresponding to the weighted *length* of the segmentation boundary [28]. Another popular choice is fully connected CRF with Gaussian kernel, a.k.a. dense CRF [130], the property of which as a regularization term is very different from that of Grid CRF [231].

The regularization term in (4.1) is pairwise. Higher-order regularization such as P^n Potts [120], curvature [173], shape prior [57], and volume prior [90] have been proposed in the literature. In general, regularization is widely used for segmentation and other computer vision tasks.

4.1.2 Regularization for Semi-supervised Learning

Roughly speaking, there are three types of machine learning, namely supervised learning, unsupervised learning, and semi-supervised learning. Semi-supervised learning [265, 241, 44] is the task of learning from labeled data and unlabeled data. Given M labeled data $(x_i, y_i) \in (\mathcal{X}, \mathcal{Y})$, $i = 1, \dots, M$ and U unlabeled data $x_i, i = M + 1, \dots, M + U$, we learn the mapping function $f(x) : \mathcal{X} \rightarrow \mathcal{Y}$. Here, we consider semi-supervised classification, i.e., $\mathcal{Y} = \{1, \dots, K\}$.

Fig. 4.1 (b) gives a toy example of binary classification with both labeled data and unlabeled data. The test point (?) is closer to negative samples. However, it looks “closer” to positive class since the data points form clusters and it is natural to assume that the decision boundary lies in region of low density. There are certain assumptions made for semi-supervised learning to work [44], some of which are equivalent to each other.

- *smoothness assumption*: If two input points x_i, x_j in high-density region are close or similar, their output prediction y_i, y_j should be similar;
- *cluster assumption*: Data points in the same cluster should have similar output prediction;
- *low density separation* Decision boundary should lie in region of low density;
- *manifold assumption* The original data $x_i \in \mathcal{X}$ is in some low-dimensional manifold.

There are many methods for semi-supervised learning, such as generative model [175], transductive support vector machine, graph-based method [24, 264, 262, 15] etc. We are motivated by graph-based method that typically minimizes the following joint objective,

$$\sum_{i=1}^M \ell(f(x_i), y_i) + \lambda \sum_{ij \in \mathcal{N}} w_{ij} \cdot [f(x_i) \neq f(x_j)], \quad (4.2)$$

where $\ell(f(x_i), y_i)$ is any standard loss function between prediction $f(x_i)$ and the ground truth label y_i , \mathcal{N} denotes edges on a graph, and w_{ij} is the similarity between x_i and x_j . The first term is the standard loss for supervised learning. The second term encourages a pair of points that are similar to have the same output prediction, which corresponds to the smoothness assumption. It softly enforces output consistency among all points based on predefined pairwise affinities $W = [w_{ij}]$.

Formulations like (4.2) with graph-based regularization are studied in several seminal work of semi-supervised learning [264, 15, 241]. These work differ in terms of *transductive* learning or *inductive* learning, the choice of the mapping function $f(x)$, and the corresponding optimization. For example, Weston et al. [241] is the first to minimize a graph regularized loss for neural networks. In other words, $f(x)$ is a neural network and the problem is to find network parameters that minimize (4.2)¹. Typical regularized semi-supervised loss function over the network output combines two terms

- *Fidelity* of network output to the labeled data
- *Regularization* of the entire network output

The purpose of the regularization term is to propagate the empirical losses over partially labeled input to the entire training data including unlabeled points.

Note that, various forms of regularization are widely used in machine learning and neural networks in particular, see Sec.4.2.3 for an overview. This paper is focused specifically on regularized losses for semi-supervised learning with partially labeled training data. In this case regularization is directly applied to the network output [241, 20]² rather than to the network parameters.

Remark. *Pairwise regularization appears both in (4.1) and (4.2) correspondingly for weakly-supervised segmentation and semi-supervised learning. It gives boundary smoothness for segmentation and low density separation for semi-supervised learning. Nowadays standard pairwise regularization is independently developed in the computer vision and the machine learning community.*

¹In [241], (4.2) is relaxed to be differentiable so as to be minimized by back-propagation.

²In general, semi-supervision losses may also regularize intermediate network layers [241, 20].

4.1.3 Our Contributions

As discussed in Sec. 4.1.1 and Sec. 4.1.2, regularization is a standard principle for segmentation and semi-supervised learning. Following the general idea of integrating (shallow) regularizers into a semi-supervised loss for (deep) learning [44, 241, 20], we advocate this principled approach in the context of weakly-supervised CNN segmentation. That is, we propose semi-supervised training loss on CNN output to combine empirical risk (e.g. cross entropy) over labeled pixels and a regularizer on all pixels. These regularization are standard in “shallow” segmentation or semi-supervised learning. A simple example is the pairwise regularization in (4.1) and (4.2). Indeed, our regularized loss is similar to the scheme in [241] regularizing network output. However, we are the first to utilize ideas in such a principled framework for weakly-supervised CNN segmentation.

Our main contributions are:

- We propose regularized loss for weakly-supervised CNN segmentation. It combines partial cross entropy loss on labeled pixels and regularization for unlabeled pixels. Regularized loss is a principled method in semi-supervised learning in deep learning [241, 20]. However, it is largely overlooked in weakly-supervised CNN segmentation dominated by training from fake but full proposals.
- Our regularized loss framework is general, allowing different regularizers in shallow segmentation as losses, for example, pairwise MRF energy [30], kernel clustering criterion [211, 222], volume constraint, etc. We propose and evaluate several regularized losses to demonstrate their effectiveness.
- We show that even without regularization, our partial cross entropy loss for scribbles (*loss sampling*) works surprisingly well compared to cross entropy over “generated” full masks that make the network over-fit to mistakes.
- Any differentiable regularized loss can be minimized by back-propagation. We give an efficient implementation of graph-based regularization with a dense graph, such as the popular Dense CRF [130].
- As an alternative to gradient descent (GD), we propose a splitting technique, alternating direction method (ADM), for minimizing regularized losses during network training. ADM can directly employ efficient regularization solvers, such as graph cut [31], known in shallow segmentation.

- Comprehensive experiments (Sec. 4.5) with our regularized losses show state-of-the-art performance for weakly supervised CNN segmentation reaching near full-supervision accuracy. While most of the experiments are about weak supervision with scribbles, regularized loss also helps CNN segmentation with image-level tags as we have shown. We also study regularized loss for fully supervised segmentation and semi-supervised segmentation (a mix of full and weak annotation).

This Chapter is structured as follows. In Sec. 4.2, we discuss the background and related work. Sec. 4.3 presents our regularized loss framework. We propose different examples of regularization originally developed in shallow segmentation or semi-supervised learning literature. In Sec. 4.4, we show an Alternative Direction Method (ADM) for the optimization of regularized losses. It can utilize effective solver for segmentation problems and works much better than gradient descent, the default optimization method for a neural network. As shown in Sec. 4.5, thorough experiments of various setting show that our regularized loss approach achieved state-of-the-art in weakly supervised CNN segmentation.

4.2 Related Work

4.2.1 CNN for Semantic Segmentation

For semantic segmentation, all leading methods in PASCAL VOC 2012 [74] train some *fully convolutional networks* (FCN) based on given ground-truth segmentations. FCN is a particular type of CNN for dense prediction problems like segmentation. Long *et al.* [150] got rid of the fully connected layers from the CNN for classification and upsampled feature maps to the original spatial resolution to enable dense semantic segmentation. Upsampling was achieved by deconvolution layer [150] and skip-connections were added to fuse low-level features and high-level features.

Strided convolution gives down-sized feature maps. Dilated convolution [250](or atrous convolution in the terminology of [45]) gives feature maps of the same size as input to the layer, yet it's receptive field grows exponentially when stacking dilated convolution layers. However, dilated convolution may give checkerboard artifact due to the dilated kernel.

Low level features represent edge, texture and other fine details while high level features represent semantics and context. It is important to combine low level and high level features for segmentation. Typically segmentation networks are of encoder-decoder type structure

with skip-connection between low resolution and high resolution features. The state-of-the-art networks are build using modules like pyramid pooling [46, 259], inception [216], residual network [102], gated multi-scale aggregation [66], and context encoding [257].

Regularization techniques for CNN segmentation include post-processing (e.g. dense CRF [130]) and appended trainable layers (e.g. CRF-as-RNN [260], Bilateral Solver [11]). For instance, DeepLab [45] popularized dense CRF as a post-processing step. In fully supervised setting, integrating the unary scores of a CNN classifier and the pairwise potentials of dense CRF achieve competitive performances [7]. This is facilitated by fast mean-field inference techniques for dense CRF based on high-dimensional filtering [3].

We study regularized losses including for example (relaxations of) standard CRF potentials. Though common as shallow regularizers [31, 30, 200, 130] or jointly trained in CNN [260, 207, 148, 7], CRF were never used directly as losses in segmentation. Our regularized loss is very different from post-processing or trainable regularization layers.

Note that optimization of CNN loss is different from that of shallow segmentation. For CNN, we minimize pixel-wise cross entropy loss using back-propagation, and the variables are network parameters. In MRF or clustering approach, we typically optimize over segmentation variables using, e.g. graph cut [31] or spectral method [211]. An interesting topic is how to leverage these optimization methods for CNN segmentation, as is explored in Sec. 4.4.

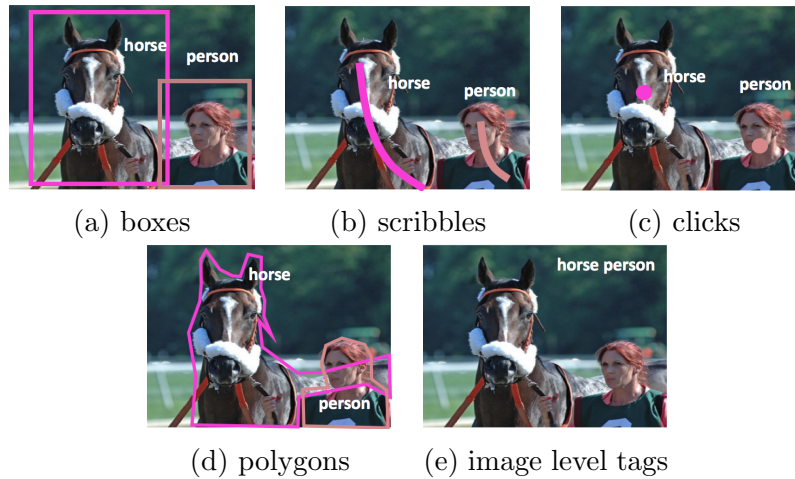


Figure 4.2: Different forms of weak annotations for training semantic segmentation.

4.2.2 Weakly-supervised CNN Segmentation

Training CNN segmentation with weak annotations has caught much interest recently. There are various interfaces of weak supervision including bounding boxes [116, 58, 244, 180], scribbles [147, 244, 232], clicks [13], polygons [38] or image-level tags [244, 180, 121], see example annotations in Fig. 4.2. Weak annotations are much faster to obtain compared to full labeling.

Weakly supervised semantic segmentation is commonly addressed by mimicking full supervision via synthesizing fully-labeled training masks (proposals) from the available partial inputs [192, 180, 147] or other weak annotations. For instance, we can first run GrabCut to obtain full labeling and then train a CNN with such proposals, see example proposals in Fig. 4.3.

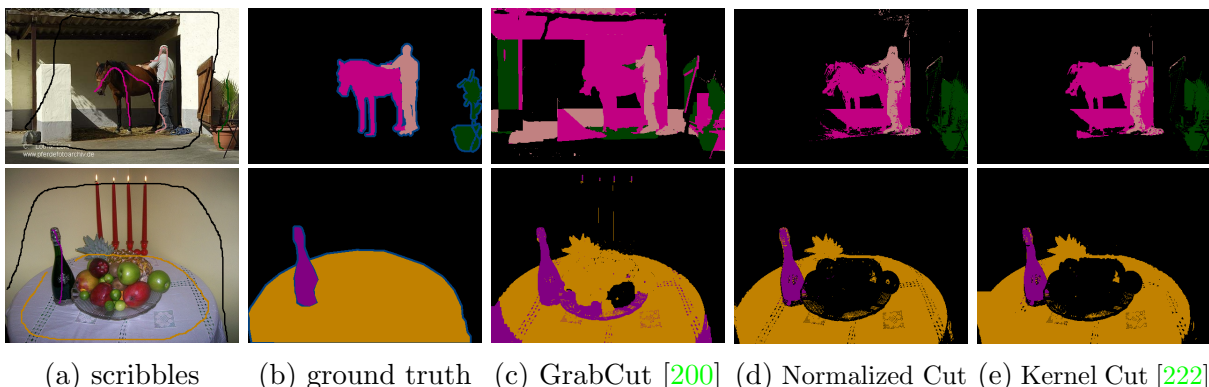


Figure 4.3: Proposals from interactive segmentation algorithms with seeds.

Why not train with segmentation proposal? It is true that many off-line interactive segmentation algorithms exist [200, 146, 191]. However, segmentation proposals have significant limitations. In practice, “shallow” segmentation proposals are likely to be erroneous, see Fig. 4.3. Most interactive segmentation methods don’t consider semantic cue. As such the proposals are misleading for training. Instead of generating unreliable proposals and train models to fit errors, our method is more direct, incorporating standard segmentation regularizer as a loss. Also heuristic pre-processing is not favored in semi-supervised learning. Modern semi-supervised learning approaches minimize a regularized loss with e.g. SVM or neural networks [15, 241], as discussed in Sec. 4.1.2.

The proposals can be generated as pre-processing or get updated during training. Prior work is typically two staged and may iterate between the two stages. One stage is proposal generation via regularization-based shallow interactive segmentation, e.g. graph cut

[147] or dense CRF mean-field inference [192, 180]. Another stage is training CNN with generated proposals. For example, Dai *et al.* [58] combine automatic segmentation proposals with bounding box annotation. The method iterates between proposal generation and network training. ScribbleSup [58] iteratively generate proposals/masks via graph cuts, while Kolesnikov *et al.* [121] used additional CRF inference layers to produce them. Papandreou *et al.* [180] proposed an Expectation-Maximization algorithm to handle weak supervision setups. Given box annotation, Khoreva *et al.* [116] utilized objectness cues and box-based GrabCut to obtain partial labeling as training data. To further simplify annotation, a recent method [13] addresses point based supervision: annotators only point to an object if it exists. In this case, additional losses are proposed, using generic objectness measure. Pathak *et al.* [184] considered cardinality constraints to generate explicit full proposals/masks.

In contrast to these two staged approaches, our method is in one stage and avoids explicit inference steps by integrating shallow regularizers directly into the loss functions. Also, iterative refinement of proposals along with network training [147, 121] is presented merely as a heuristics in prior work. We show connections between proposal generation and our regularized loss framework in Sec. 4.4.

Xu *et al.* [244] formulated all types of weak supervision as linear constraints in max-margin clustering. This framework is somewhat flexible and principled. However, we take advantage of deep CNN rather than SVM used in [244].

Among different forms of annotations, the most challenging is with image level tags. In this case, localization cues can be extracted from a classification network, i.e., class activation mapping (CAM) [261]. For example, Kolesnikov *et al.* [121] trained segmentation network from partial labeling that comes from thresholded CAM.

4.2.3 Regularization for Neural Networks

Regularizations have also been widely used in neural networks to avoid over-fitting or encourage sparsity, e.g. Norm regularization [89], Dropout [213] and ReLU [85]. Norm regularization penalizes parameters' magnitude to limit model complexity. Dropout [213] regularizes the activations through random masking, which implicitly realize model ensemble. ReLU [85] is a sparsity-inducing regularization, see [89] for more regularization examples.

Our regularized loss differs from these regularization. Ours is a semi-supervised loss for regularizing network output for unlabeled data. Such regularization is well coupled with partial fidelity loss, allowing implicit label propagation during training.

4.3 Our Regularized Losses

This section introduces our regularized losses for weakly-supervised segmentation. In general, the use of regularized losses is a well-established approach in semi-supervised deep learning [241, 89]. We advocate this principle for semantic CNN segmentation, propose specific shallow regularizers for such losses, and discuss their properties.

Assuming image I and its *partial* ground truth labeling or mask Y , let $f_\theta(I)$ be the output of a segmentation network parameterized by θ . In general, CNN training with our joint regularized loss corresponds to optimization problem of the following form

$$\min_{\theta} \ell(f_\theta(I), Y) + \lambda \cdot R(f_\theta(I)) \quad (4.3)$$

where $\ell(S, Y)$ is a ground truth loss and $R(S)$ is a regularization term or regularization loss. Both losses have argument $S = f_\theta(I) \in [0, 1]^{|\Omega| \times K}$, which is K -way softmax segmentation generated by a network. Using cross entropy over partial labeling as the ground truth loss, we have the following joint *regularized semi-supervised loss*

$$\sum_{p \in \Omega_{\mathcal{L}}} H(Y_p, S_p) + \lambda \cdot R(S) \quad (4.4)$$

where $\Omega_{\mathcal{L}} \subset \Omega$ is the set of labeled pixels and $H(Y_p, S_p) = -\sum_k Y_p^k \log S_p^k$ is the cross entropy between network predicted segmentation $S_p \in [0, 1]^K$ (a row of matrix S corresponding to point p) and ground truth labeling $Y_p \in \{0, 1\}^K$.

In principle, any function $R(S)$ can be used as a loss given its gradient or sub-gradient. This paper studies (relaxations of) regularizers from shallow segmentation as loss functions. Section 4.3.1 details our MRF/CRF loss and its implementation. In Section 4.3.2, we propose *kernel cut loss* combining CRF with kernel clustering criterion and justify this combination. Sec. 4.3.3 shows a simple loss for incorporating linear constraint.

Remark. *The partial cross entropy loss only considers the cross entropy loss for labeled pixels $p \in \Omega_{\mathcal{L}}$ which effectively ignores other regions. We are not the first to ignore regions in weakly-supervised segmentation in general, as there are examples for boxes [116] and for clicks [13]. However, this partial loss can be seen as a sampling of the loss. In practice, we found that training only with this simple loss works surprisingly well, achieving more than 85% of the accuracy compared to using the full labeling. In fact, using this loss is even better than training from GrabCut proposals as we show in our experiments in Sec. 4.5 but this trick has been overlooked in previous work [147]. This supports our argument in Sec. 4.2.2 that segmentation proposals may be very misleading.*

4.3.1 MRF Energy as Loss

Assuming that segmentation variables S_p are restricted to binary class indicators $S_p \in \{0, 1\}^K$, the standard Potts/CRF model [31] could be represented via Iverson brackets $[\cdot]$, as on the left hand side below

$$\sum_{p,q \in \Omega} w_{pq} [S_p \neq S_q] = \sum_{p,q \in \Omega} w_{pq} \|S_p - S_q\|^2, \quad (4.5)$$

where $W = [w_{pq}]$ is a matrix of pairwise discontinuity costs or an *affinity matrix*. The right hand side above is a particularly straightforward quadratic relaxation of the Potts model that works for relaxed $S_p \in [0, 1]^K$ corresponding to a typical soft-max output of CNNs. In fact, this quadratic function is very common in the general context of regularized weakly supervised losses in deep learning [241].

As discussed in the introduction, this relaxation is not unique, e.g. TV-based [40] and convex formulations [43, 189], L_p relaxations [54], LP and other relaxations [63, 226]. Evaluation of different relaxations in the context of regularized weak supervision losses is left for future work. We use slightly different quadratic relaxation of the Potts model

$$R_{CRF}(S) = \sum_k S^{k'} W (\mathbf{1} - S^k) \quad (4.6)$$

expressed in terms of support vectors for each label k , i.e. columns of the segmentation matrix $S^k \in [0, 1]^{|\Omega|}$. For discrete segment indicators (4.6) gives the cost of a cut between segments, same as the Potts model on the left hand side of (4.5), but it differs from the relaxation of the right hand side of (4.5).

The affinity matrix W can be sparse or dense. Sparse W commonly appears in the context of boundary regularization and edge alignment in shallow segmentation [30]. With dense Gaussian kernel W_{pq} (4.6) is a relaxation of DenseCRF [130].

To implement $R_{CRF}(S)$ (4.6) as a loss, we need to compute its gradient w.r.t. S^k ,

$$\frac{\partial R_{CRF}(S)}{\partial S^k} = -2WS^k. \quad (4.7)$$

For sparse W , (4.7) is fast to compute. For DenseCRF where W is fully connected Gaussian, computing the gradient (4.7) becomes a standard Bilateral filtering problem, for which many fast methods were proposed [3, 182]. We implement our loss layers using fast Gaussian filtering [3], which is also utilized in the inference of DenseCRF [130, 260, 207]. Note that our CRF loss layer is much faster than CRF inference layer [121, 260] since iterative mean-field is not needed.

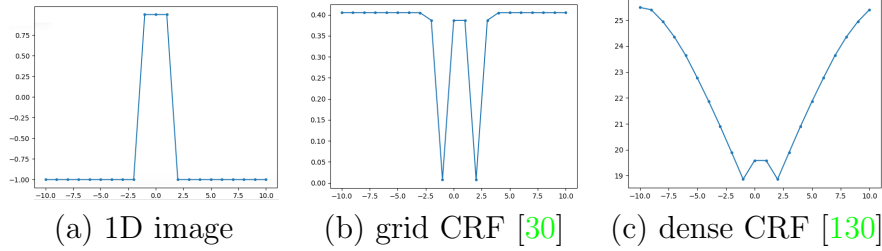


Figure 4.4: *Synthetic segmentation example* for grid and dense CRF (Potts) models: (a) intensities $I(x)$ on 1D image. The cost of segments $S^t = \{x \mid x < t\}$ with different discontinuity points t according to (b) nearest-neighbor (grid) Potts and (c) larger-neighborhood (dense) Potts. The latter gives smoother cost function, but its flatter minimum may complicate discontinuity localization.

Grid CRF vs Dense CRF The affinities w_{pq} can be on sparse grid or densely connected, leading to Grid CRF or Dense CRF. The nearest-neighbor version over k -connected grid \mathcal{N}_k , as well as its popular variational analogues, *e.g.* geodesic active contours [37], convex relaxations [189, 41], or continuous max-flow [254], are particularly well-researched. It is common to use contrast-weighted discontinuity penalties [31, 30] between the neighboring points, as emphasized by the condition $\{pq\} \in \mathcal{N}_k$ below

$$w_{pq} = \exp \frac{-\|I_p - I_q\|^2}{2\sigma^2} \cdot [\{pq\} \in \mathcal{N}_k]. \quad (4.8)$$

Nearest neighbor Potts models minimize the contrast-weighted length of the segmentation boundary preferring shorter perimeter aligned with image edges, *e.g.* see Fig. 4.5(b). The popularity of this model can be explained by generality, robustness, well-established foundations in geometry, and a large number of efficient discrete or continuous solvers that guarantee global optimum in binary problems [30] or some quality bound in multi-label settings, *e.g.* α -expansion [31].

Dense CRF [130] is a Potts model where pairwise interactions are active over significantly bigger neighborhoods defined by a Gaussian kernel with a relatively large bandwidth Δ over pixel locations

$$w_{pq} = \exp \frac{-\|I_p - I_q\|^2}{\sigma^2} \cdot \exp \frac{-\|p - q\|^2}{\Delta^2}. \quad (4.9)$$

Its use in shallow vision is limited as it often produces noisy boundaries [130], see also Fig. 4.5(c). Also, global optimization methods mentioned above do not scale to dense



Figure 4.5: *Real "shallow" segmentation example* for sparse (b) and dense (c) CRF (Potts) models for image with seeds (a). Sparse Potts gives smoother segment boundary with better edge alignment, while dense CRF inference often gives noisy boundary.

neighborhoods. Yet, dense CRF model is popular in the context of CNNs where it can be used as a differentiable regularization layer [260, 207]. Larger bandwidth yields smoother objective (3.2), see Fig. 4.4(c), amenable to gradient descent or other local linearization methods like mean-field inference that are easy to parallelize. Note that existing efficient inference methods for dense CRF require *bilateral filtering* [130], which is restricted to Gaussian weights as in (4.9). This is in contrast to global Potts solvers, *e.g.* α -expansion, that can use arbitrary weights, but become inefficient for dense neighborhoods.

Noisier dense CRF results, *e.g.* in Fig. 4.5(c), imply weaker regularization. Indeed, as discussed in [231], for larger neighborhoods the Potts model gets closer to *cardinality potentials*. Bandwidth Δ in (4.9) is a resolution scale at which the model sees the segmentation boundary. Weaker regularization in dense CRF may preserve some thin structures smoothed out by fine-resolution boundary regularizers, *e.g.* nearest-neighbor Potts. However, this is essentially the same “noise preservation” effect shown in Fig. 4.5(c). For consistency, the rest of the paper refers to the nearest-neighbor Potts model as *grid CRF*, and large-neighborhood Potts as *dense CRF*.

4.3.2 Clustering Criterion as Loss [219]

Besides the CRF loss (4.6), we also propose its combination with clustering criterion, for example normalized cut for image segmentation [211]. Normalized Cut is a popular graph clustering algorithm originally proposed for image segmentation [211]. It is the sum of ratios between the cuts and the volumes. We relax to continuous vector S^k and our normalized cut loss [219] is as follows,

$$R_{NC}(S) = \sum_k \frac{S^{k'} \hat{W} (1 - S^k)}{d' S^k} \stackrel{c}{=} \sum_k -\frac{S^{k'} \hat{W} S^k}{d' S^k}. \quad (4.10)$$

where $d = \hat{W}\mathbf{1}$ are node degrees. Its gradient w.r.t. S^k is,

$$\frac{\partial R_{NC}(S)}{\partial S^k} = \frac{S^{k'} \hat{W} S^k d}{(d' S^k)^2} - \frac{2 \hat{W} S^k}{d' S^k}. \quad (4.11)$$

Combining CRF and NC Loss

The combined *kernel cut loss* is simply a linear combination of (4.6) and (4.10)

$$R_{KC}(S) = \sum_k S^{k'} W (1 - S^k) + \gamma \sum_k \frac{S^{k'} \hat{W} (1 - S^k)}{d' S^k} \quad (4.12)$$

which is motivated by *kernel cut* shallow segmentation [222] with complementary benefits of balanced normalized cut clustering and object boundary regularization or edge alignment as in Potts model. Accordingly, the gradient of (4.12) is the linear combination of (4.7) and (4.11).

Remark. *As objectives or regularization for "shallow" segmentation techniques, both normalized cut and Potts/CRF are very popular. It is not hard to see that CRF loss (4.6) is the cut for discrete labeling. Normalized cut differs from pairwise CRF by having extra normalization. Such normalization was originally motivated for balanced unsupervised segmentation [211]. Though CRF and normalized cut only differ by having normalization or not, we argue that their clustering / regularization effect is different. Normalized cut is known to be equivalent to kernel K-means [64] that encourages balanced non-linear color clustering, while CRF was primarily motivated for edge alignment in segmentation. Combining pairwise CRF and high-order normalized cut as losses, possibly with different affinity matrix W and \hat{W} , gives extra degree of freedom during training. In our experiments, the best weakly supervised segmentation is achieved with kernel cut loss.*

Note that standard normalized cut and CRF objectives in shallow segmentation require fairly different optimization techniques (e.g. spectral relaxation or graph cuts), but the standard gradient descent approach for optimizing losses during CNN training allows significant flexibility in including different regularization terms, as long as there is a reasonable relaxation.

4.3.3 Linear Constraint as Loss

Often, we have inequality constraint for segmentation. An example is the volume constraint. We may know the lower bound a_k and the upper bound b_k of the size of particular

segment S^k , which is quite common for medical images [172, 91]. Also, in the case of weakly-supervised segmentation with image tags [184, 121], the existing class should have positive size, i.e., $|S^k| \geq 1$. This means we need to solve a constrained optimization problem with volume constraint.

$$\begin{aligned} \min_{\theta} \quad & \ell(f_{\theta}(I), Y) \\ \text{s.t.} \quad & a_k \leq |S^k| \leq b_k, \forall k. \end{aligned} \tag{4.13}$$

Though the constraint is linear (and convex), the loss is highly nonconvex w.r.t. network parameters. Standard Lagrange dual based approach is intractable for the problem above with linear constraint. We need to optimize the whole CNN for all training images in each step of dual descent. We propose to add a penalty term directly to the loss if the constraint is not satisfied.

$$R(S^k) = \begin{cases} (|S^k| - a_k)^2, & \text{if } |S^k| < a_k \\ (|S^k| - b_k)^2, & \text{if } |S^k| > b_k \\ 0, & \text{otherwise.} \end{cases} \tag{4.14}$$

Such a simple loss would be zero if the size of the segment $|S^k|$ is in the range of $[a_k, b_k]$, and the gradient computation is trivial. Despite its simplicity, we find our volume loss (4.13) works well in practice for training CNN segmentation with weak supervision. Indeed, it outperforms the previous method [184] which is much more complicated. Pathak *et al.* [184] introduced latent variables and minimized a cross entropy loss between latent segmentation (proposals) and network output segmentation, while the proposals are sought with linear constraint. Our regularized loss method is more straightforward and efficient for incorporating such linear constraint.

4.4 Beyond Gradient Descent for Regularized Losses

The simplicity of gradient descent (GD) made it the default method for training ever-deeper and complex neural networks. Both loss functions and architectures are often explicitly tuned to be amenable to this basic local optimization. In the context of weakly-supervised CNN segmentation, we demonstrate a well-motivated loss function where an alternative optimizer (ADM) achieves the state-of-the-art while GD performs poorly. Interestingly, GD obtains its best result for a “smoother” tuning of the loss function. The

results are consistent across different network architectures. Our loss is motivated by well-understood MRF/CRF regularization models in “shallow” segmentation and their known global solvers. Our work suggests that network design/training should pay more attention to optimization methods.

4.4.1 Alternative Direction Method (ADM)

As an alternative to gradient descent (GD), we propose a splitting technique, *alternating direction method* (ADM)³, for minimizing *regularized losses* during network training. ADM can directly employ efficient regularization solvers known in shallow segmentation. Compared to GD, our ADM approach with α -expansion solver significantly improves optimization quality for the *grid CRF* (nearest-neighbor Potts) loss in weakly supervised CNN segmentation.

We present a general alternating direction method (ADM) to optimize regularized losses (4.4), which is equivalent to the following problem,

$$\begin{aligned} \min_{\theta} \quad & \sum_{p \in \Omega_{\mathcal{L}}} H(Y_p, S_p) + \lambda \cdot R(X) \\ \text{s.t.} \quad & X = S \end{aligned} \tag{4.15}$$

where $X = \{X_p \in \{0, 1\}^k\}$ are introduced hidden variables. To solve the constrained optimization problem (4.15), we further introduce divergence measure D , *e.g.* the Kullback-Leibler divergence between X and S_{θ} . $R(X)$ can now be a standard *discrete* MRF regularization, *e.g.* (3.2). Now, we solve the following dual problem augmented with $D(X|S)$, which is unconstrained.

$$\max_{\gamma} \min_{\theta, X} \sum_{p \in \Omega_{\mathcal{L}}} H(Y_p, S_p) + \lambda R(X) + \gamma \sum_{p \in \Omega_{\mathcal{U}}} D(X_p | S_p) \tag{4.16}$$

We alternate optimization over X and θ in (4.16). The dual update, *i.e.* maximization over γ increases its value at every update resulting in a variant of simulated annealing. We have experimented with variable multiplier γ but found no advantage compared to fixed γ . So, we fix γ and do not optimize for it. In summary, instead of optimizing the regularization term with gradient descent, our approach splits regularized-loss problem

³Standard ADMM [26] casts a problem $\min_x f(x) + g(x)$ into $\min_{x,y} \max_{\lambda} f(x) + g(y) + \lambda^T(x - y) + \rho\|x - y\|^2$ and alternates updates over x , y and λ optimizing f and g in parallel. Our ADM uses a different form of splitting and can be seen as a *penalty method*.

(4.4) into two sub-problems. We replace the softmax outputs $S_{p,\theta}$ in the regularization term by latent discrete variables X_p and ensure consistency between both variables (*i.e.* , S_θ and X) by minimizing divergence D .

This is similar conceptually to the general principles of ADMM [26]. Our ADM splitting accommodates the use of powerful and well-established discrete solvers for the regularization loss. As we show in Sec. 4.5, the popular α -expansion solver [31] significantly improves optimization of grid CRF losses yielding state-of-the-art training quality. Such efficient solvers guarantee global optimum in binary problems [30] or a quality bound in multi-label case [31].

Our discrete-continuous ADM method alternates two steps, each decreasing (4.16), until convergence. Given fixed discrete latent variables X_p computed at the previous iteration, the first step learns the network parameters θ by minimizing the following loss via standard back-propagation and a variant of stochastic gradient descent (SGD):

$$\min_{\theta} \sum_{p \in \Omega_{\mathcal{L}}} H(Y_p, S_p) + \gamma \sum_{p \in \Omega_{\mathcal{U}}} D(X_p | S_p) \quad (4.17)$$

The second step fixes the network output S_θ and finds the next latent binary variables X by minimizing the following objective over X via any suitable discrete solver:

$$\begin{aligned} \min_{X \in \{0,1\}^{|\Omega| \times K}} \quad & \lambda R(X) + \gamma \sum_{p \in \Omega_{\mathcal{U}}} D(X_p | S_p) \\ \text{s.t.} \quad & X_p = Y_p \quad \forall p \in \Omega_{\mathcal{L}}. \end{aligned} \quad (4.18)$$

Because X_p is a discrete variable with only K possible values, the second term in (4.18) is a basic unary term. Similarly, the equality constraints could be implemented as unary terms using prohibitive values of unary potentials. Unary terms are simplest possible energy potentials that can be handled by any general discrete solver. On the other hand, the regularization term $R(X)$ usually involves interactions of two or more variables introducing new properties of solution together with optimization complexity. In case of grid CRF as $R(\cdot)$, one can use graph cut [30], α -expansion [31], QPBO [198], TRWS [124] *etc.*

In summary, our approach alternates the two steps described above. For each minibatch we compute network prediction, then compute hidden variables X optimizing (4.18), then compute gradients of loss (4.17) and update the parameters of the network using a variant of SGD.

4.5 Experiments

Our main contribution is regularized loss for weakly-supervised CNN segmentation. In Sec. 4.5.1, we show visualization of gradients of these losses, which drives the network to learn. Sec. 4.5.2 is the main experimental result of this paper. We minimize different regularized losses with gradient descent and compare quantitatively and qualitatively the segmentation obtained. For weakly-supervised segmentation with scribbles [147], we train using different regularized losses including our proposed CRF loss, high-order normalized cut loss, and kernel cut loss, as discussed in Sec. 4.3. We show that combining CRF (4.6) with normalized cut (4.10) *a la* KernelCut [222] yields the best performance. Our method achieved the state-of-the-art for weakly supervised segmentation with scribbles. We also train with shortened scribbles to see how much each method degrades.

CRF regularization has been combined with CNN segmentation in the form of post-processing, trainable layers, or proposal generation in training. In Sec. 4.5.3, we compare these schemes with our regularized loss approach in the context of weakly-supervised segmentation. Besides for scribbles, we also utilize our regularized loss framework for image-level labels based supervision and compare to SEC [121], a recent method based on proposal generation.

Regularized loss can be optimized by stochastic gradient descent (GD) or alternative direction method (ADM), as discussed in Sec. 4.4. In Sec. 4.5.4, we compare three training schemes, namely dense CRF with GD, grid CRF with GD and grid CRF with ADM for weakly supervised CNN segmentation. We also investigate regularization loss for fully or semi-supervised segmentation (with labeled and unlabeled images), see Sec. 4.5.5.

Dataset: Most experiments are on the PASCAL VOC12 segmentation dataset. For all method, we train with the augmented dataset of 10,582 images. The scribble annotations are from [147]. Following standard protocol, mean intersection-over-union (mIoU) is evaluated on the *val* set that contains 1,449 images. For image-level label supervision, our experiment setup is the same as in [121].

Implementation details: Our implementation is based on DeepLab v2 [45]. We follow the learning rate strategy in DeepLab v2 ⁴ for the baseline with full supervision. For our method of regularized loss, we first train with partial cross entropy loss only. Then we fine-tune with extra regularized losses of different types. Our CRF and normalized cut regularization losses are defined at full image resolution. If the network outputs shrunk labeling, which is typical, the labeling is interpolated to original size before feeding into the loss layer.

⁴<https://bitbucket.org/aquariusjay/deeplab-public-ver2>

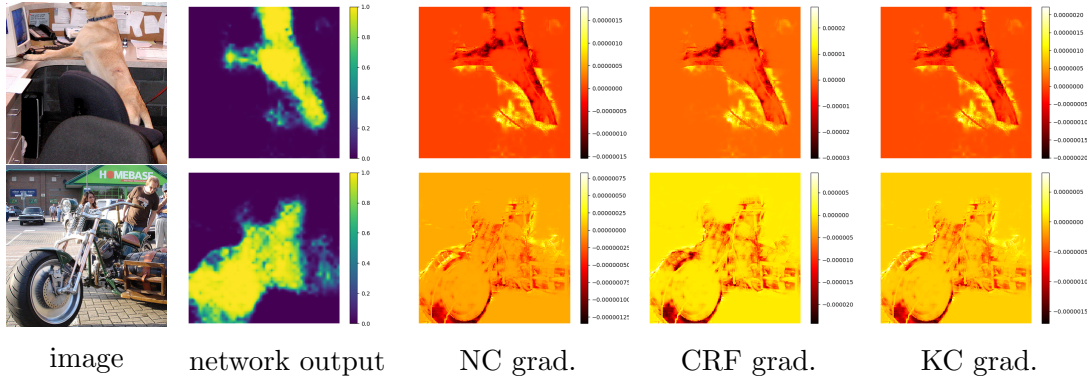


Figure 4.6: Visualization of the gradient for different losses. The negative (positive) gradients are coded in red (yellow). For example, negative gradients on the dog drives the network to predict “dog” for these pixels. Also, the dog pops out in the gradient map.

4.5.1 Visualization of Gradients

To get some intuition about these losses and their regularization effect, we visualize their gradient w.r.t. segmentation $\frac{\partial R(S)}{\partial S}$ in Fig. 4.6. Note that the *sign* of gradients indicates whether to encourage or discourage certain labeling. The color coded gradients clearly show evidence toward better color clustering and edge alignment for normalized cut and CRF. The gradients of different losses are slightly different and complement each other. Since kernel cut is the combination of normalized cut with CRF, then its gradient is the sum of that of each.

4.5.2 Comparison of Regularized Losses

Our regularized loss framework is general and allows integration with multiple regularizations. We are interested in the effect of each term in our joint loss, including partial cross entropy, normalized cut, and CRF regularization. We conducted ablation study and trained the networks with different combinations of the losses.

Tab. 4.5 summaries the results with different regularized losses. Here we report both result on various networks. The baselines are with cross entropy losses of full labeled masks or partial seeds ignoring unlabeled region. We choose the weight of the regularization term to achieve the best validation accuracy. Consistently over different networks, using the proposed CRF loss outperforms that with the normalized cut loss. Our best result is obtained when combining both normalized cut loss and DenseCRF loss. Clearly, utilization

	Weak				Full
	CE only	w/ NC	w/ Dense CRF	w/ KernelCut	
DeepLab-MSc-largeFOV	56.0 (8.1)	60.5 (3.6)	63.1 (1.0)	63.5 (0.6)	64.1
DeepLab-VGG16	60.4 (8.4)	62.4 (6.4)	64.4 (4.4)	64.8 (4.0)	68.8
DeepLab-ResNet101	69.5 (6.1)	72.8 (2.8)	72.9 (2.7)	73.0 (2.6)	75.6

Table 4.1: mIOU on PASCAL VOC2012 *val* set. Our flexible framework allows various types of regularization losses for weakly supervised segmentation, e.g. normalized cut, CRF or their combinations (KernelCut [222]) as joint loss. We achieved the state-of-the-art with scribbles. In () shows the offset to the result with full masks.

of CRF loss and KernelCut loss reduce the gap toward the full supervision baseline. With DeepLab-MSc-largeFOV, using KernelCut regularized loss achieved mIOU of 63.5%, while previous best is 60.5% with normalized cut loss [219]. Our result with scribbles approaches **63.5/64.1=99.0%** of the quality of that with full supervision, yet only **3%** of all pixels are scribbled. This work pushes the limit of weakly supervised segmentation.

Fig. 4.7 shows some qualitative examples with different losses. Our Kernel Cut loss combines the benefit of both regional color clustering (normalized cut) and pairwise regularization (DenseCRF). By combining both we can achieve better segmentation regularization.

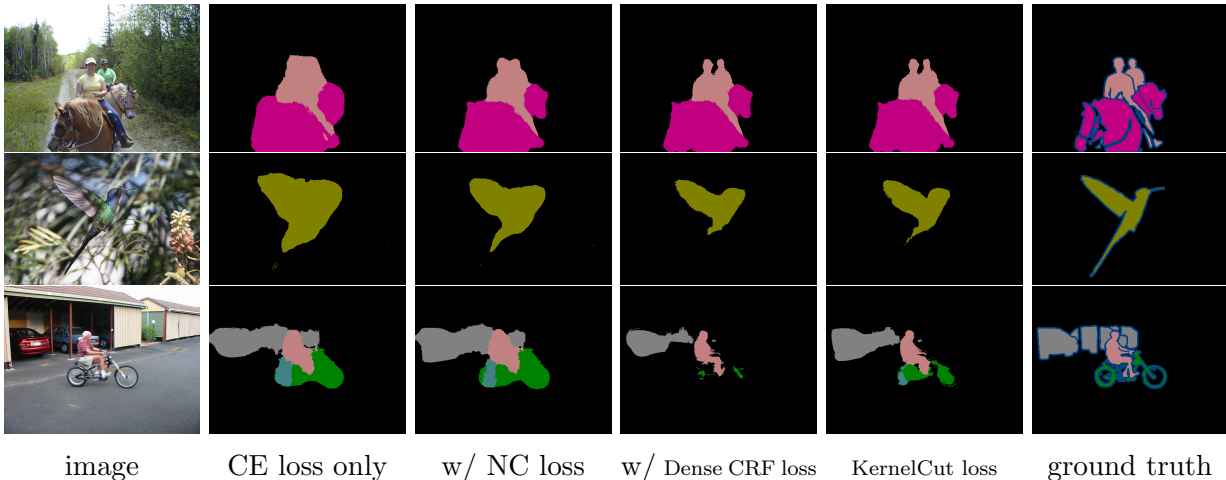


Figure 4.7: Examples on PASCAL VOC *val* set. Kernel cut as regularization loss gives qualitatively better result than that with normalized cut loss. We found kernel cut results to have better edge alignment.

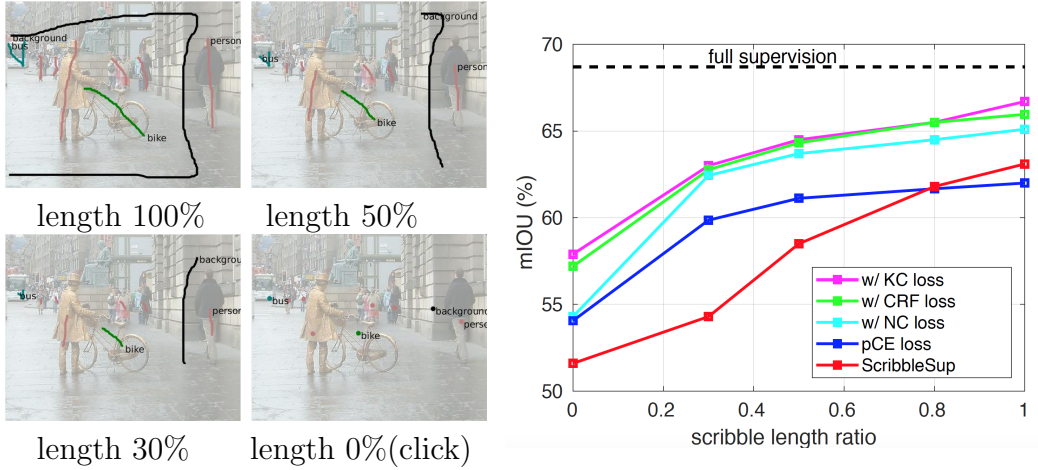


Figure 4.8: Similar to [147], we shorten the scribbles. With length zero (clicks) is the most challenging case. Right plot shows mIOUs when train with shorter scribbles.

Train with shortened scribbles

To see the limit of our algorithm with scribble supervision, we train with shortened scribbles visualized in Fig. 4.8. Note that with length zero, there is only one click for each object. For different length ratios from zero to 100%, our regularized loss method achieved much better segmentation than ScribbleSup [147]. The improvement over ScribbleSup [147] is more significant for shorter scribbles, in which case ScribbleSup gives low quality proposals.

4.5.3 CRF as Loss, Post-processing or Trainable Layers

We are the first to propose CRF loss though it’s popular to have CRF as post-processing [45] or jointly trained with the network [260, 207]. For example, CRF-as-RNN [260] is proposed for fully supervised segmentation. Here for weakly-supervised segmentation with scribbles, we train CRF-as-RNN but only minimizing partial cross entropy loss on scribbles. Table 4.2 compares the effects of CRF as loss, post-processing or trainable layers. End-to-end training of CRF helps a little bit (64.8% vs 64.3%), but the best is achieved with our CRF loss, which is also much more efficient without any recurrent inference. Note that the plain network trained with extra CRF loss is even better than a network trained without such loss but followed by CRF post-processing, see the fourth and second row in Table 4.2 (64.4% vs 64.3%). This shows the effectiveness of our CRF loss for training CNN segmentation.

training	testing	mIOU (%)
partial cross entropy loss	plain network	60.4
partial cross entropy loss	disjoint network and CRF	64.3
partial cross entropy loss end-to-end CRF	jointly trained network and CRF	64.8
partial cross entropy loss and our CRF loss	plain network	64.4
partial cross entropy loss and our CRF loss	disjoint network and CRF	66.4

Table 4.2: Ablation study of CRF as loss, post-processing [45] or trainable layers [260] for weakly supervised segmentation with scribbles for DeepLab-VGG16.

We compare our DenseCRF loss to the constrain-to-boundary loss in SEC [121]. In the forward pass, SEC generates segmentation proposals by running iterative mean-field inference of DenseCRF for which the unaries are network output. Then in the backward pass, the proposals are regarded as fake ground-truth and the cross entropy loss w.r.t. full proposals is minimized. We show that such iterative scheme of DenseCRF inference and network learning is redundant. Our DenseCRF loss serves the purpose of regularizing network output during training.

		include this loss?			
Losses	Seeding loss [121]	✓	✓	✓	✓
	Expansion loss [121]	✓	✓	✓	✓
	Constrain-to-boundary loss [121]		✓	*	
	Our CRF loss				✓
mIOU (%)		38.4	43.7	43.8	43.9
Overall training time in s/batch		0.86	1.19 (0.33)	1.19 (0.33)	0.98 (0.12)

Table 4.3: Tag-based weak supervision. We train with different combinations of the losses in SEC [121] and our CRF loss. Replacing the constrain-to-boundary loss in SEC [121] by CRF loss gives minor improvement in accuracy, but training with regularized loss with gradient descent is faster since no iterative CRF inference is needed. We also compare to a variant (*) of SEC without back-propagation of the CRF inference layer. Parenthesis (·) show the time for the constrain-to-boundary loss layer or our direct loss layer.

As mentioned earlier, SEC [121] was originally focused on tag-based supervision. Table 4.3 reports some tests for that form of weak supervision. We compare SEC with its

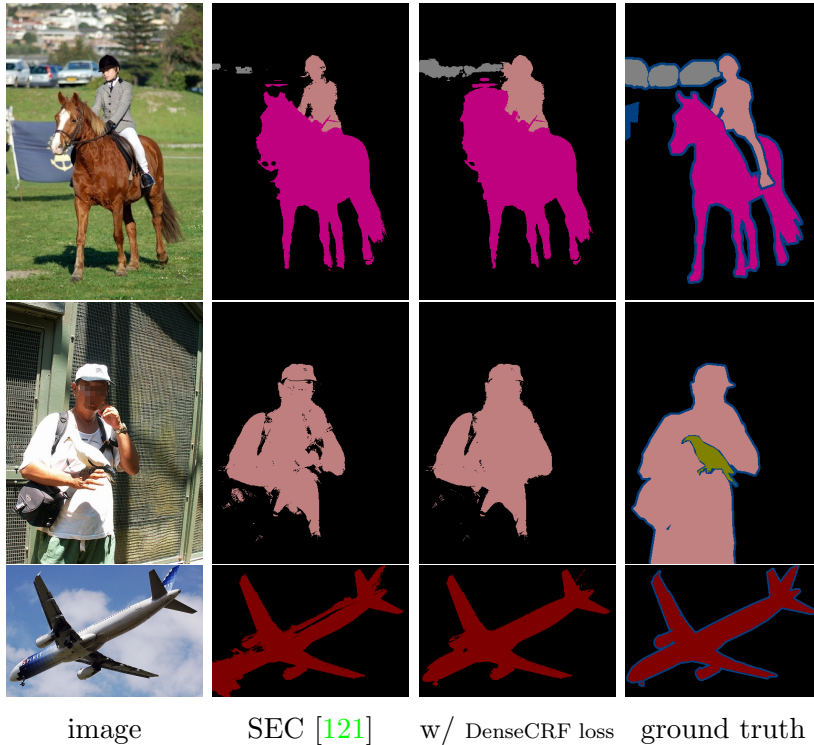


Figure 4.9: Examples for supervision with image-level labels (tags). We train using the seeding loss, expansion loss in SEC [121] and our CRF loss. Similar segmentation is obtained yet we avoid any iterative mean-field inference for dense CRF.

simplification replacing their constrain-to-boundary loss by our regularized loss. We train using different combinations of losses for supervision based on image-level labels/tags. Our CRF loss helps to improve training to 43.9% compared to 38.4% without it. There is only small improvement in mIOU when replacing constrain-to-boundary loss by CRF loss.

However, our CRF loss layer is several times faster than constrain-to-boundary layer integrating explicit iterative inferences. The segmentation accuracy and overall training speed are also reported in Tab. 4.3. (The results are for the DeepLab-largeFOV network.) We also tested a variant of SEC without back-propagation of mean-field layer, which we show is not helping in practice. Fig. 4.9 shows testing examples for our method and SEC with image tags as supervision.

network	training set [†]		validation set	
	GD	ADM	GD	ADM
Deeplab-LargeFOV	2.52	2.41	2.51	2.33
Deeplab-MSc-largeFOV	2.51	2.40	2.49	2.33
Deeplab-VGG16	2.37	2.10	2.42	2.14
Resnet-101	2.66	2.49	2.61	2.42

Table 4.4: ADM gives better grid CRF losses than gradient descend (GD). [†]We randomly selected 1,000 training examples.

4.5.4 GD vs ADM for Grid CRF Loss

We test both the grid CRF and dense CRF as regularized losses. Such regularized loss can be optimized by stochastic gradient descent (GD) or alternative direction method (ADM), as discussed in Sec. 4.4. We compare three training schemes, namely dense CRF with GD [224], grid CRF with GD and grid CRF with ADM.

Loss Minimization

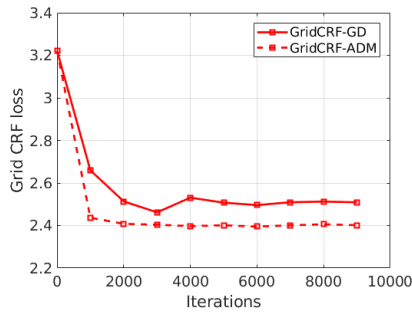


Figure 4.10: Training progress of ADM and gradient descent (GD) on Deeplab-MSc-largeFOV. Our ADM for the grid CRF loss with α -expansion significantly improves convergence and achieves lower training loss. For example, first 1,000 iterations of ADM give grid CRF loss lower than GD’s best result.

Here, we show that for grid CRF losses the ADM approach employing α -expansion [31], a powerful discrete optimization method, outperforms common gradient descent methods for regularized losses [219, 224] in terms of finding a lower minimum of regularization

loss. Tab. 4.4 shows the grid CRF losses on both training and validation sets for different network architectures.

Fig. 4.10(a) shows the evolution of the grid CRF loss over the number of iterations of training. ADM requires fewer iterations to achieve the same CRF loss. The networks trained using ADM scheme give lower CRF losses for both training and validation sets. Thus, in the context of grid CRFs, the ADM approach coupled with α -expansion shows drastic improvement in the optimization quality. In the next section, we further compare ADM with GD to see which gives better segmentation.

Segmentation Quality

The quantitative measures for segmentation by different methods are summarized in Tab. 4.5 and Tab. 4.6. The mIOU and segmentation accuracy on the *val* set of PASCAL 2012 [74] are reported for various networks. The supervision is scribbles [147]. The quality of weakly supervised segmentation is bounded by that with full supervision and we are interested in the gap for different weakly supervised approaches.

The baseline approach is to train the network using proposals generated by GrabCut style interactive segmentation with such scribbles. Besides the baseline (train w/ proposals), here we compare variants of regularized losses optimized by gradient descent or ADM. The regularized loss is comprised of the partial cross entropy (pCE) w.r.t. scribbles and grid/dense CRF. Other losses *e.g.* normalized cut [211, 219] may give better segmentation, but the focus is to compare gradient descent vs ADM optimization for the grid CRF.

It is common to apply dense CRF post-processing [45] to the network’s output during testing. However, for the sake of clear comparison, we show results without it.

As shown in Tab. 4.5, all regularized approaches work better than the non-regularized approach that only minimizes the partial cross entropy. Also, the regularized loss approaches are much better than proposal generation based method since erroneous proposals may mislead training.

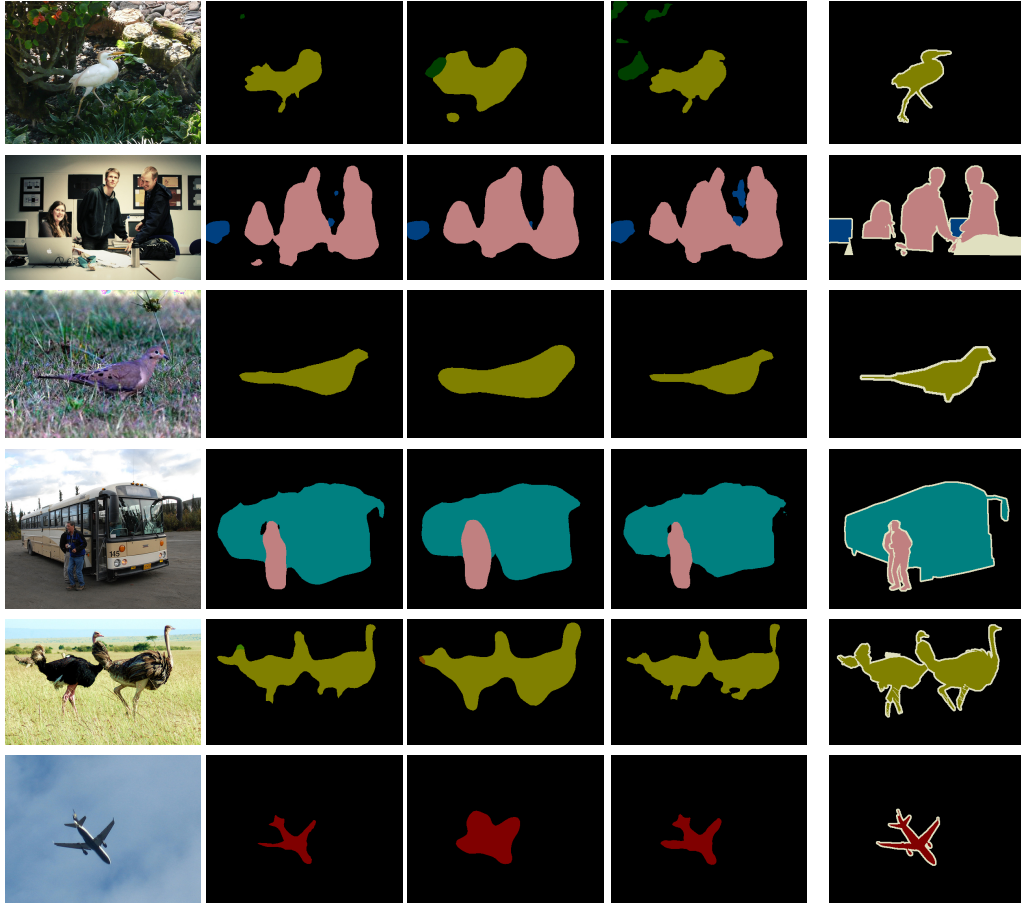
Among regularized loss approaches, grid CRF with GD performs the worst due to the fact that a first-order method like gradient descent leads to the poor local minimum for the grid CRF in the context of energy minimization. Our ADM for the grid CRF gives much better segmentation competitive with the dense CRF with GD. The alternative grid CRF based method gives good quality segmentation approaching that for full supervision. Tab. 4.6 shows *accuracy* of different methods for pixels close to the semantic boundaries. Such measure tells the quality of segmentation in boundary regions. Fig. 4.11 shows a few qualitative segmentation results.

Network	Full sup.	Weak supervision				
		train w/ proposals	pCE loss	+dense CRF loss	+grid CRF loss	
				GD [224]	GD	ADM
Deeplab-largeFOV	63.0	54.8	55.8	62.2	60.4	61.7
Deeplab-MSc-largeFOV	64.1	55.5	56	63.1	61.2	62.9
Deeplab-VGG16	68.8	59.0	60.4	64.4	63.3	65.2
ResNet-101	75.6	64.0	69.5	72.9	71.7	72.8

Table 4.5: Weakly supervised segmentation results for different choices of network architecture, regularized losses and optimization via gradient descent or ADM. We show mIOU on *val* set of PASCAL 2012. ADM consistently improves over GD for different networks for grid CRF. Our grid CRF with ADM is competitive to previous state-of-the-art dense CRF (with GD) [224].

	Network	Full sup.	Weak supervision				
			train w/ proposals	pCE loss	+Dense CRF loss	+grid CRF loss	
					GD [224]	GD	ADM
all pixels	Deeplab-MSc-largeFOV	90.9	86.4	86.5	90.6	89.9	90.5
	Deeplab-VGG16	91.6	88.6	88.9	91.1	90.5	91.3
	ResNet-101	94.5	90.2	92	93.1	92.9	93.4
trimap 16 pixels	Deeplab-MSc-largeFOV	80.1	73.9	66.7	77.8	74.8	76.7
	Deeplab-VGG16	81.9	75.5	70.9	77.8	75.6	78.1
	ResNet-101	85.7	78.4	77.7	82.0	80.6	82.2
trimap 8 pixels	Deeplab-MSc-largeFOV	75.0	69.5	60.3	72.5	68.4	71.4
	Deeplab-VGG16	76.9	70.4	64.1	72.0	69.0	72.4
	ResNet-101	81.5	73.8	71.2	76.7	74.6	77.0

Table 4.6: Pixel-wise accuracy on *val* set of PASCAL 2012. Top 3 rows: accuracy over all pixels. Middle 3 rows: accuracy for pixels within 16 pixels away from semantic boundaries. Bottom 3 rows: accuracy for pixels within 8 pixels away from semantic boundaries. Pixels closer to boundaries are more likely to be mislabeled. Our ADM scheme improves over GD for grid CRF loss consistently for different networks. Note that weak supervision with our approach is almost as good as full supervision.



(a) input (b) Dense GD (c) Grid GD (d) Grid ADM (e) ground truth

Figure 4.11: Example segmentations (Deeplab-MSc-largeFOV) by variants of regularized loss approaches. Gradient descent (GD) for grid CRF gives segmentation of poor boundary alignment though grid CRF is part of the regularized loss. ADM for grid CRF significantly improves edge alignment and compares favorably to dense CRF based method.

NC loss weight	mIOU	cross entropy loss	NC loss
0	89.85%	0.106	0.536
0.1	89.38%	0.110	0.517
0.2	89.39%	0.112	0.509
0.5	88.75%	0.125	0.485

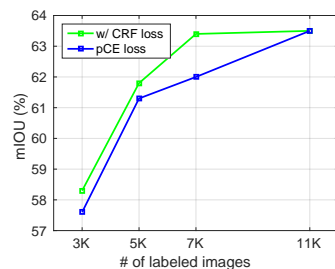
Table 4.7: Negative effect of regularization loss for full supervision.

4.5.5 Fully Supervision and Semi-supervision

We’ve demonstrated the usefulness of regularized loss for weakly supervised segmentation. Here we test if it also helps full supervision or semi-supervision.

Full supervision: We add NC loss on fully labeled images besides the cross entropy loss. This experiment is on a simple saliency dataset [47] where color clustering is obvious and likely to help. As shown in Tab. 4.7, when we increase the weight of $R_{NC}(S)$, we indeed obtained segmentation that is more regularized. However, with extra regularization loss during training, the cross entropy loss got worse and mIOU decreased. The conclusion is that imposing regularized loss naively on labeled images doesn’t help. Empirical risk minimization is in some sense optimal for fully labeled data. Extra regularization loss steers the network in the wrong direction if the regularization doesn’t totally agree with the ground truth.

Semi supervision: For training with both labeled images and unlabeled images, our joint losses include cross entropy on labeled images and regularization on unlabeled ones. We drop the labeling for some of the the 11K images in PASCAL VOC 2012. We train DeepLab-LargeFOV with different amount of labeled & unlabeled images, see right plot. For the baseline that can only utilize labeled images, the performance degrades with less masks, as expected. For our framework, the labeled and unlabeled images are mixed and randomly sampled in each batch. We observed 0.7% 1.5% improvement with our regularized loss. Note that this result is highly preliminary. We also train with the 11K labeled images in VOC 2012 and the 10K unlabeled in VOC 2017 and achieved boosted performance from 63.5% to 64.3%.



Chapter 5

Conclusion and Future Work

This thesis presents the work I have done in image segmentation during my PhD, with particular focus on optimization. For segmentation with weak supervision or annotation, we propose new formulations combining standard techniques including Markov Random Field [23, 30, 200], Kernel Clustering [211, 235], and Convolutional Neural Networks [150, 45]. We also developed novel optimization method for the corresponding joint objectives or regularized losses. We conduct thorough experiments in lots of applications such as interactive segmentation, motion segmentation, image clustering, and semantic segmentation.

To summarize, in Chapter 1, we reviewed related techniques including MRF, Kernel Clustering, and CNN. We briefly discussed their pros and cons, which motivates combining them. Through such combination, we take advantage of these deep and non-deep methods. One of our observations is that many algorithms such as K-means [153] and GrabCut [200] can be interpreted as bound optimization, which motivates novel optimization algorithms.

In Chapter 2, we propose a new general pseudo-bound optimization paradigm for approximate iterative minimization of high-order and non-submodular MRF energies. It generalizes the standard majorize-minimize principle relaxing the bound constraint for an auxiliary function. We propose to optimize a family of pseudo-bounds at each iteration [217]. To guarantee the energy decreases, we include at least one bound in the family. We propose parametric maxflow [125] to explore all global minima for the whole family in low-order polynomial time.

In Chapter 3, we combine the two seemingly different methodologies for data clustering/partitioning: kernel clustering [211, 235, 64] and MRF regularization based segmentation [30, 23]. They differ in terms of motivation, formulation, and optimization, e.g.

spectral relaxation vs maxflow. We explain how to take advantage of the mutual benefits of MRF regularization and kernel clustering. Our joint energy combines standard regularization, e.g., MRF potentials, and kernel clustering criteria like normalized cut. Complementarity of such terms is demonstrated in many applications using our bound optimization Kernel Cut algorithm [223] for the joint energy.

In Chapter 4, we combine deep CNN with shallow non-deep techniques such as MRF regularization and Kernel clustering. The combination is achieved simply by minimizing a regularized loss for weakly-supervised CNN segmentation [224]. Regularized semi-supervised loss is a principled approach to semi-supervised deep learning, in general. We propose and study various regularization as losses, including MRF energy, Clustering criterion, or linear constraints for segmentation. In contrast to our regularized loss approach, the mainstream in weakly supervised segmentation rely on generating "fake" full masks from partial input and train a network to match the proposals. For regularized loss, we propose an optimization method beyond gradient descent [157]. Our alternating direction method (ADM) can utilize established solver for regularized segmentation, such as graph cut. Indeed, heuristic proposal based methods are related to approximate alternating direction method (ADM) for the optimization of regularized loss. Our ADM based analysis gives insights, and we show that for Grid CRF, ADM gives better optimization and segmentation than that achieved by gradient descent. We achieve the state-of-the-art in weakly supervised CNN segmentation, yielding almost the same accuracy with only a fraction of labeling during training.

However, our work here has some limitations, and it will be interesting to extend in the following ways.

- Better approximation: While our iterative bound optimization for MRF energies and Normalized Cut in Chapter 2 and Chapter 3 guarantee the decrease of the objective, the linear bound itself may not be tight enough as a good approximation. For example, the kernel bound (3.52) for normalized cut is linear. Having second order bound or even higher order bound will give better optimization. There is trade-off between the complexity and the tightness of the bound.
- Better relaxation: As a loss to be minimized by backpropagation, segmentation regularization needs to be relaxed to continuous domain $S_p \in [0, 1]^K$. It is also important how we relax discrete regularization to a continuous formulation, which affects gradient and optimization. For example in Sec. 4.3.1, the discrete pairwise CRF $\sum_{p,q \in \Omega} w_{pq} [S_p \neq S_q]$ is relaxed as $\sum_k S^{k'} W(\mathbf{1} - S^k)$. There are many alternatives, e.g. TV-based [40] and convex formulations [43, 189], L_p relaxations [54], LP and other

relaxations [63, 226]. These relaxations were proposed for MRF inference algorithms that deal with relaxed variables. It is interesting to see the effect of different relaxations for training in this context. Note that gradient descent is efficient for grid CRF since the potentials are sparse. For various ways of relaxations [63, 226], how to efficiently compute gradient for dense CRF with a full Gaussian affinity matrix W is not obvious. Besides for pairwise regularization, we will also investigate relaxation for high-order regularization, e.g. normalized cut and P^n Potts [120] model.

- Multi-tasks: In this thesis, we combine different segmentation techniques. Other computer vision tasks, such as 3D reconstruction, tracking, and depth estimation, are closely related to segmentation. The best way is to solve them jointly. For example, optical flow can help distinguish a moving object from the background, and segmentation can tell about discontinuity boundary of optical flow. Also, ground-truth for optical flow is rare, while ground-truth for segmentation is fairly more common. It is challenging to learn these perception tasks jointly across various domains and utilize labeled data from different domains.
- Multi-objectives: The work here is all about a single joint objective. Our regularized loss consists of empirical risk loss and various kinds of regularization loss. Essentially, this is a multi-objective optimization problem. Suppose we have N regularization terms $R_1(S_\theta), R_2(S_\theta), \dots, R_N(S_\theta)$. Currently, we minimize a weighted sum of the losses $\ell(S_\theta, Y) + \sum_{n=1}^N \lambda_n \cdot R_n(S_\theta)$. Then accordingly, the gradient of the regularized loss w.r.t. S_θ is the weighted sum of gradients of each term. Despite its simplicity, this approach has its limitations. Different tasks may not agree with each other. There is likely trade-off between the tasks. We propose multi-objective optimization [50, 78, 77, 208] for regularized losses in deep learning. In multi-objective optimization, the objective is a vector rather than a scalar: $\min_{\theta} (R_0(S_\theta), \dots, R_N(S_\theta))^T$. Note that, here for notation convenience, we let $R_0(S_\theta) = \ell(S_\theta, Y)$. The goal in multi-objective optimization is to find a Pareto optimal solution. We are mostly interested in multi-objective optimization for regularized loss with full-supervision, for which standard single-objective optimization gives negative results.
- Irregular data: In this work, we studied image segmentation, for which CNN has been a great success. However, CNN cannot be directly applied to point clouds or graphs, which can be obtained from the nowadays ubiquitous depth sensors. We are interested in segmentation of point clouds, graph, and mesh. The bulk of this thesis is about graph regularization and graph algorithms. It would be interesting to leverage the work here in graph clustering and regularized loss to graph neural network [118, 93, 204] for weakly-supervised graph segmentation.

References

- [1] Cplex optimizer. <http://www.cplex.com>.
- [2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Ssstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov 2012.
- [3] Andrew Adams, Jongmin Baek, and Myers Abraham Davis. Fast high-dimensional filtering using the permutohedral lattice. *Computer Graphics Forum*, 29(2):753–762, 2010.
- [4] Charu C. Aggarwal and Chandan K. Reddy, editors. *Data Clustering: Algorithms and Applications*. Chapman & Hall / CRC, 2014.
- [5] Le Thi Hoai An, Pham Dinh Tao, Nam Nguyen Canh, and Nguyen V. Thoai. Dc programming techniques for solving a class of nonlinear bilevel programs. *J. Global Optimization*, 44(3):313–337, 2009.
- [6] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):898–916, 2011.
- [7] Anurag Arnab, Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Måns Larsson, Alexander Kirillov, Bogdan Savchynskyy, Carsten Rother, Fredrik Kahl, and Philip Torr. Conditional random fields meet deep neural networks for semantic segmentation. *IEEE Signal Processing Magazine Special Issue in Deep Learning for Visual Understanding White Paper*, 2017.
- [8] I. Ben Ayed, A. Mitiche, and Z. Belhadj. Multiregion level set partitioning of synthetic aperture radar images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(5):793–800, 2005.

- [9] Francis Bach and Michael Jordan. Learning spectral clustering. *Advances in Neural Information Processing Systems*, 16:305–312, 2003.
- [10] Xue Bai, Jue Wang, David Simons, and Guillermo Sapiro. Video snapcut: robust video object cutout using localized classifiers. In *ACM Transactions on Graphics (ToG)*, volume 28, page 70. ACM, 2009.
- [11] Jonathan T Barron and Ben Poole. The fast bilateral solver. In *European Conference on Computer Vision*, pages 617–632. Springer, 2016.
- [12] Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley, 2006.
- [13] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. Whats the point: Semantic segmentation with point supervision. In *European Conference on Computer Vision*, pages 549–565. Springer, 2016.
- [14] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.
- [15] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006.
- [16] Serge Belongie and Jitendra Malik. Finding boundaries in natural images: A new method using point descriptors and area completion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1998.
- [17] I. Ben Ayed, H-M Chen, K. Punithakumar, I. Ross, and S. Li. Graph cut segmentation with a global constraint: Recovering region distribution via a bound of the bhattacharyya measure. In *CVPR*, pages 3288–3295, 2010.
- [18] Ismail Ben Ayed, Lena Gorelick, and Yuri Boykov. Auxiliary cuts for general classes of higher order functionals. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1304–1311, Portland, Oregon, June 2013.
- [19] Shai Ben-david, Ulrike Von Luxburg, and Dvid Pl. A sober look at clustering stability. In *In COLT*, pages 5–19. Springer, 2006.
- [20] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

- [21] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):259–279, 1986.
- [22] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, August 2006.
- [23] Andrew Blake, Pushmeet Kohli, and Carsten Rother. *Markov random fields for vision and image processing*. Mit Press, 2011.
- [24] Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. 2001.
- [25] Endre Boros and Peter L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123:2002, 2001.
- [26] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.
- [27] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [28] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *International Conference on Computer Vision*, volume I, pages 26–33, 2003.
- [29] Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient N-D image segmentation. *International Journal of Computer Vision (IJCV)*, 70(2):109–131, 2006.
- [30] Yuri Boykov and Marie-Pierre Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *IEEE International Conference on Computer Vision (ICCV)*, number 105-112, 2001.
- [31] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, November 2001.
- [32] Leo Breiman. Technical note: Some properties of splitting criteria. *Machine Learning*, 24(1):41–47, 1996.
- [33] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

- [34] Thomas Brox and Jitendra Malik. Object segmentation by long term analysis of point trajectories. In *Computer Vision–ECCV 2010*, pages 282–295. Springer, 2010.
- [35] Joachim M. Buhmann. Information theoretic model validation for clustering. 2012.
- [36] Miguel A. Carreira-Perpinan and Weiran Wang. The K-Modes Algorithm for Clustering. In *arXiv:1304.6478v1 [cs.LG]*, April 2013.
- [37] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.
- [38] Lluís Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *CVPR*, volume 1, page 2, 2017.
- [39] Antonin Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical imaging and vision*, 20(1-2):89–97, 2004.
- [40] Antonin Chambolle and Jérôme Darbon. On total variation minimization and surface evolution using parametric maximum flows. *International Journal of Computer Vision*, 84(3):288, April 2009.
- [41] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.
- [42] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Trans. Image Processing*, 10(2):266–277, 2001.
- [43] Tony F Chan, Selim Esedoglu, and Mila Nikolova. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM journal on applied mathematics*, 66(5):1632–1648, 2006.
- [44] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.
- [45] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.
- [46] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *arXiv preprint arXiv:1802.02611*, 2018.

- [47] Ming-Ming Cheng, Niloy J. Mitra, Xiaolei Huang, Philip H. S. Torr, and Shi-Min Hu. Global contrast based salient region detection. *IEEE TPAMI*, 37(3):569–582, 2015.
- [48] Selene E. Chew and Nathan D. Cahill. Semi-supervised normalized cuts for image segmentation. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [49] Radha Chitta, Rong Jin, Timothy Havens, and Anil Jain. Scalable kernel clustering: Approximate kernel k-means. In *KDD*, pages 895–903, 2011.
- [50] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. *Evolutionary algorithms for solving multi-objective problems*, volume 5. Springer, 2007.
- [51] Maxwell D Collins, Ji Liu, Jia Xu, Lopamudra Mukherjee, and Vikas Singh. Spectral clustering with a convex regularizer on millions of images. In *Computer Vision–ECCV 2014*, pages 282–298. Springer, 2014.
- [52] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.
- [53] Dorin Comaniciu and Peter Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(5):603–619, 2002.
- [54] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot. A unifying graph-based optimization framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1384–1399, July 2011.
- [55] I. J. Cox, S. B. Rao, and Y. Zhong. "ratio regions": A technique for image segmentation. In *Proceedings of the 13th International Conference on Pattern Recognition - Volume 2*, ICPR '96, pages 557–, Washington, DC, USA, 1996. IEEE Computer Society.
- [56] Trevor Cox and Michael Cox. *Multidimensional scaling*. CRC Press, 2000.
- [57] Daniel Cremers, Mikael Rousson, and Rachid Deriche. A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape. *International journal of computer vision*, 72(2):195–215, 2007.

- [58] Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1635–1643, 2015.
- [59] A Philip Dawid. Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and computing*, 2(1):25–36, 1992.
- [60] Andrew DeLong, Anton Osokin, Hossam Isack, and Yuri Boykov. Fast approximate energy minimization with label costs. *International Journal of Computer Vision*, 96:1–27, Jan. 2012.
- [61] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. Deepglobe 2018: A challenge to parse the earth through satellite images. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 172–17209. IEEE, 2018.
- [62] Zhuo Deng, Sinisa Todorovic, and Longin Jan Latecki. Semantic segmentation of rgb-d images with mutex constraints. In *International Conference on Computer Vision (ICCV)*, Santiago, Chile, December 2015.
- [63] Alban Desmaison, Rudy Bunel, Pushmeet Kohli, Philip HS Torr, and M Pawan Kumar. Efficient continuous relaxations for dense crf. In *European Conference on Computer Vision*, pages 818–833. Springer, 2016.
- [64] Inderjit Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means, spectral clustering and normalized cuts. In *KDD*, 2004.
- [65] Inderjit Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors: A multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Learning (PAMI)*, 29(11):1944–1957, November 2007.
- [66] Henghui Ding, Xudong Jiang, Bing Shuai, Ai Qun Liu, and Gang Wang. Context contrasted feature and gated multi-scale aggregation for scene segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [67] Mingsong Dou, Jonathan Taylor, Henry Fuchs, Andrew Fitzgibbon, and Shahram Izadi. 3d scanning deformable objects with a single rgb-d sensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 493–501, 2015.

- [68] RC Dubes, AK Jain, SG Nadabar, and CC Chen. Mrf model-based algorithms for image segmentation. In *[1990] Proceedings. 10th International Conference on Pattern Recognition*, volume 1, pages 808–814. IEEE, 1990.
- [69] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, 2001.
- [70] Sebastien Roy and Ingemar J Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *IEEE Proceedings of International Conference on Computer Vision (ICCV'98), Bombay*, 1998.
- [71] Mark J. Eisner and Dennis G. Severance. Mathematical techniques for efficient record segmentation in large shared databases. *J. ACM*, 23(4):619–635, October 1976.
- [72] Noha El-Zehiry and Leo Grady. Fast global optimization of curvature. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, number 3257–3264, 2010.
- [73] Anders Eriksson, Carl Olsson, and Fredrik Kahl. Normalized cuts revisited: A reformulation for segmentation with linear grouping constraints. *Journal of Mathematical Imaging and Vision*, 39(1):45–61, 2011.
- [74] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.
- [75] Mark Fashing and Carlo Tomasi. Mean shift is a bound optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:471–474, 2005.
- [76] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient belief propagation for early vision. *International journal of computer vision*, 70(1):41–54, 2006.
- [77] Joerg Fliege, LM Grana Drummond, and Benar Fux Svaiter. Newton’s method for multiobjective optimization. *SIAM Journal on Optimization*, 20(2):602–626, 2009.
- [78] Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, 2000.
- [79] S Fraclik. Learning to recognize patterns without a teacher. *IEEE Transactions on Information Theory*, 13(1):57–64, 1967.

- [80] Brendan J Frey and David JC MacKay. A revolution: Belief propagation in graphs with cycles. In *Advances in neural information processing systems*, pages 479–485, 1998.
- [81] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.*, 18(1):30–55, February 1989.
- [82] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.
- [83] Mark Girolami. Mercer kernel-based clustering in feature space. *IEEE Trans. Neural Networks*, 13(3):780–784, 2002.
- [84] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.
- [85] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.
- [86] Andrew V Goldberg and Robert E Tarjan. A new approach to the maximum-flow problem. *Journal of the ACM (JACM)*, 35(4):921–940, 1988.
- [87] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.
- [88] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.
- [89] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
- [90] L. Gorelick, Y. Boykov, O. Veksler, I. Ben Ayed, and A. Delong. Submodularization for binary pairwise energies. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, Ohio, June 2014.
- [91] L. Gorelick, F. R. Schmidt, and Y. Boykov. Fast trust region for segmentation. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1714–1721, Portland, Oregon, June 2013.

- [92] Lena Gorelick, Olga Veksler, Yuri Boykov, and Claudia Nieuwenhuis. Convexity shape prior for segmentation. In *European Conference on Computer Vision*, pages 675–690. Springer, 2014.
- [93] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 729–734. IEEE, 2005.
- [94] Jens-Malte Gottfried, Janis Fehr, and Christoph S Garbe. Computing range flow from multi-modal kinect data. In *Advances in Visual Computing*, pages 758–767. Springer, 2011.
- [95] Stephen Gould. Max-margin learning for lower linear envelope potentials in binary markov random fields. In *ICML*, 2011.
- [96] John C Gower and Gavin JS Ross. Minimum spanning trees and single linkage cluster analysis. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 18(1):54–64, 1969.
- [97] Leo Grady. Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1768–1783, 2006.
- [98] Dorothy M Greig, Bruce T Porteous, and Allan H Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society: Series B (Methodological)*, 51(2):271–279, 1989.
- [99] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.
- [100] Varun Gulshan, Victor Lempitsky, and Andrew Zisserman. Humanising grabcut: Learning to segment humans using the kinect. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1127–1133. IEEE, 2011.
- [101] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.
- [102] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

- [103] Matthias Hein, Thomas Navin Lal, and Olivier Bousquet. Hilbertian metrics on probability measures and their application in svms. *Pattern Recognition*, LNCS 3175:270–277, 2004.
- [104] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Brian Kingsbury, et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29, 2012.
- [105] Dorit S. Hochbaum. Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):889–898, 2010.
- [106] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [107] T. Hofmann and J. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 19(1):1–14, January 1997.
- [108] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2462–2470, 2017.
- [109] Hiroshi Ishikawa. Exact optimization for Markov Random Fields with convex priors. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003.
- [110] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. Kernel methods on Riemannian manifolds with Gaussian RBF kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 37(12):2464–2477, 2015.
- [111] I.H. Jermyn and H. Ishikawa. Globally optimal regions and boundaries as minimum ratio weight cycles. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(10):1075–1088, 2001.
- [112] Hao Jiang. Linear solution to scale invariant global figure ground separation. In *CVPR*, pages 678–685, 2012.

- [113] Jörg H Kappes, Bjoern Andres, Fred A Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X Kausler, Thorben Kröger, Jan Lellmann, et al. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision*, 115(2):155–184, 2015.
- [114] Michael Kearns, Yishay Mansour, and Andrew Ng. An Information-Theoretic Analysis of Hard and Soft Assignment Methods for Clustering. In *Conf. on Uncertainty in Artificial Intelligence (UAI)*, August 1997.
- [115] Hoel Kervadec, Jose Dolz, Meng Tang, Eric Granger, Yuri Boykov, and Ismail Ben Ayed. Constrained-cnn losses for weakly supervised segmentation. *Medical image analysis*, 54:88–99, 2019.
- [116] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, Honolulu, HI, USA, 2017.
- [117] Junhwan Kim, Vladimir Kolmogorov, and Ramin Zabih. Visual correspondence using energy minimization and mutual information. In *Int. Conf. on Comp. Vision (ICCV)*, pages 1033 – 1040, vol.2, October 2003.
- [118] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.
- [119] Maria Klodt and Daniel Cremers. A convex framework for image segmentation with moment constraints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2236–2243. IEEE, 2011.
- [120] Pushmeet Kohli, L’Ubor Ladický, and Philip H. Torr. Robust higher order potentials for enforcing label consistency. *Int. J. Comput. Vision*, 82(3):302–324, May 2009.
- [121] Alexander Kolesnikov and Christoph H. Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *European Conference on Computer Vision (ECCV)*. Springer, 2016.
- [122] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. 2009.

- [123] V. Kolmogorov and T. Schoenemann. Generalized sequential tree-reweighted message pass. In *arXiv:1205.6352*, 2012.
- [124] Vladimir Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, October 2006.
- [125] Vladimir Kolmogorov, Yuri Boykov, and Carsten Rother. Applications of parametric maxflow in computer vision. In *IEEE International Conference on Computer Vision (ICCV)*, 2007.
- [126] Vladimir Kolmogorov and Carsten Rother. Minimizing non-submodular functions with graph cuts - a review. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 29(7):1274–1279, July 2007.
- [127] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.
- [128] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, February 2004.
- [129] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf optimization via dual decomposition: Message-passing revisited. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.
- [130] Philipp Krahenbuhl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *NIPS*, 2011.
- [131] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [132] Brian Kulis, Sugato Basu, Inderjit Dhillon, and Raymond Mooney. Semi-supervised graph clustering: a kernel approach. *Machine Learning*, 74(1):1–22, January 2009.
- [133] M Pawan Kumar, Vladimir Kolmogorov, and Philip HS Torr. An analysis of convex relaxations for map estimation of discrete mrfs. *Journal of machine learning research*, 10(Jan):71–106, 2009.

- [134] Abhijit Kundu, Yin Li, Frank Dellaert, Fuxin Li, and James M Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. In *European Conference on Computer Vision*, pages 703–718. Springer, 2014.
- [135] Lubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip H. S. Torr. Graph cut based inference with co-occurrence statistics. In *Proceedings of the 11th European Conference on Computer Vision: Part V, ECCV’10*, pages 239–253, Berlin, Heidelberg, 2010. Springer-Verlag.
- [136] Kenneth Lange, David R. Hunter, and Ilsoon Yang. Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, 9(1):1–20, 2000.
- [137] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [138] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [139] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2000.
- [140] Victor Lempitsky, Andrew Blake, and Carsten Rother. Image segmentation by branch-and-mincut. In *ECCV*, 2008.
- [141] Victor Lempitsky, Pushmeet Kohli, Carsten Rother, and Toby Sharp. Image segmentation with a bounding box prior. In *Int. Conference on Computer Vision (ICCV)*, pages 277–284, 2009.
- [142] Victor Lempitsky, Carsten Rother, Stefan Roth, and Andrew Blake. Fusion moves for markov random field optimization. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1392–1405, 2010.
- [143] Marius Leordeanu, Martial Hebert, and Rahul Sukthankar. An integer projected fixed point method for graph matching and map inference. In *NIPS*, pages 1114–1122, 2009.
- [144] S.Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer-Verlag, 3rd edition, 2009.

- [145] Tao Li, Sheng Ma, and Mitsunori Ogihara. Entropy-based criterion in categorical clustering. In *Proc. of Intl. Conf. on Machine Learning (ICML)*, pages 536–543, 2004.
- [146] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Lazy snapping. In *ACM Transactions on Graphics (ToG)*, volume 23, pages 303–308. ACM, 2004.
- [147] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3159–3167, 2016.
- [148] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *Proceedings of the IEEE international conference on computer vision*, pages 1377–1385, 2015.
- [149] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.
- [150] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.
- [151] Gilles Louppe, Louis Wehenkel, Antonio Sutera, and Pierre Geurts. Understanding variable importances in forests of randomized trees. In *NIPS*, pages 431–439, 2013.
- [152] David J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.
- [153] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.
- [154] Michael Maire, Takuya Narihira, and Stella X. Yu. Affinity CNN: Learning pixel-centric pairwise relations for figure/ground embedding. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [155] Jitendra Malik, Serge Belongie, Thomas Leung, and Jianbo Shi. Contour and texture analysis for image segmentation. *International journal of computer vision*, 43(1):7–27, 2001.

- [156] Kevis-Kokitsi Maninis, Sergi Caelles, Jordi Pont-Tuset, and Luc Van Gool. Deep extreme cut: From extreme points to object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 616–625, 2018.
- [157] Dmitrii Marin, Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Beyond Gradient Descent for Regularized Segmentation Losses. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [158] Dmitrii Marin, Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Kernel clustering: Breiman’s bias and solutions. *IEEE transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(1):136–147, January 2019.
- [159] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 416–423. IEEE, 2001.
- [160] Kevin McGuinness and Noel E O’connor. A comparative evaluation of interactive segmentation algorithms. *Pattern Recognition*, 43(2):434–444, 2010.
- [161] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [162] Anton Milan, Laura Leal-Taixé, Konrad Schindler, and Ian Reid. Joint tracking and segmentation of multiple targets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5397–5406, 2015.
- [163] Amar Mitiche and Ismail Ben Ayed. *Variational and Level Set Methods in Image Segmentation*. Springer, 2010.
- [164] Lopamudra Mukherjee, Vikas Singh, and Chuck R. Dyer. Half-integrality based algorithms for cosegmentation of images. In *CVPR*, pages 2028–2035, 2009.
- [165] Klaus-Robert Müller, Sebastian Mika, Gunnar Rätsch, Koji Tsuda, and Bernhard Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.
- [166] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.

- [167] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4293–4302, 2016.
- [168] M. Narasimhan and J. A. Bilmes. A submodular-supermodular procedure with applications to discriminative structure learning. In *UAI*, pages 404–412, 2005.
- [169] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.
- [170] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.
- [171] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: analysis and an algorithm. In *Advances in neural information processing systems (NIPS)*, volume 2, pages 849–856, 2002.
- [172] Marc Niethammer and Christopher Zach. Segmentation with area constraints. *Medical image analysis*, 17(1):101–112, 2013.
- [173] C. Nieuwenhuis, E. Toeppe, L. Gorelick, O. Veksler, and Y. Boykov. Efficient squared curvature. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, Ohio, June 2014.
- [174] Claudia Nieuwenhuis and Daniel Cremers. Spatially varying color distributions for interactive multilabel segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 35(5):1234–1247, 2013.
- [175] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2):103–134, 2000.
- [176] Peter Ochs and Thomas Brox. Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1583–1590. IEEE, 2011.
- [177] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *IEEE transactions on pattern analysis and machine intelligence*, 36(6):1187–1200, 2014.

- [178] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.
- [179] Gabriel L Oliveira, Wolfram Burgard, and Thomas Brox. Efficient deep models for monocular road segmentation. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 4885–4891. IEEE, 2016.
- [180] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1742–1750, 2015.
- [181] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. In *Computer Vision–ECCV 2006*, pages 568–580. Springer, 2006.
- [182] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. *International journal of computer vision*, 81(1):24–52, 2009.
- [183] Kyoungup Park and Stephen Gould. On learning higher-order consistency potentials for multi-class pixel labeling. In *ECCV*, 2012.
- [184] Deepak Pathak, Philipp Krähenbühl, and Trevor Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1796–1804, 2015.
- [185] Massimiliano Pavan and Marcello Pelillo. Dominant sets and pairwise clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):167–172, 2007.
- [186] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
- [187] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.
- [188] Viet-Quoc Pham, Keita Takahashi, and Takeshi Naemura. Foreground-background segmentation using iterated distribution matching. In *CVPR*, pages 2113–2120, 2011.

- [189] Thomas Pock, Antonine Chambolle, Daniel Cremers, and Horst Bischof. A convex relaxation approach for computing minimal partitions. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [190] Renfrey Burnard Potts. Some generalized order-disorder transformations. In *Mathematical proceedings of the cambridge philosophical society*, volume 48, pages 106–109. Cambridge University Press, 1952.
- [191] Brian L Price, Bryan Morse, and Scott Cohen. Geodesic graph cut for interactive image segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3161–3168. IEEE, 2010.
- [192] Martin Rajchl, Matthew CH Lee, Ozan Oktay, Konstantinos Kamnitsas, Jonathan Passerat-Palmbach, Wenjia Bai, Mellisa Damodaram, Mary A Rutherford, Joseph V Hajnal, Bernhard Kainz, et al. Deepcut: Object segmentation from bounding box annotations using convolutional neural networks. *IEEE transactions on medical imaging*, 36(2):674–683, 2017.
- [193] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.
- [194] Xiaofeng Ren, Liefeng Bo, and Dieter Fox. Rgb-(d) scene labeling: Features and algorithms. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2759–2766. IEEE, 2012.
- [195] Kenneth Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of the IEEE*, 86(11):2210–2239, 1998.
- [196] Stefan Roth and Michael J Black. Fields of experts. *International Journal of Computer Vision*, 82(2):205, 2009.
- [197] Volker Roth, Julian Laub, Motoaki Kawanabe, and Joachim Buhmann. Optimal cluster preserving embedding of nonmetric proximity data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(12):1540–1551, 2003.
- [198] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary mrfs via extended roof duality. In *IEEE CVPR*, 2007.

- [199] C. Rother, T. P. Minka, A. Blake, and V. Kolmogorov. Cosegmentation of image pairs by histogram matching - incorporating a global constraint into mrfs. In *CVPR*, pages 993–1000, 2006.
- [200] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut - interactive foreground extraction using iterated graph cuts. In *ACM tran. on Graphics (SIG-GRAPH)*, 2004.
- [201] M. Rousson and Deriche R. A variational framework for active and adaptative segmentation of vector valued images. In *Workshop on Motion and Video Computing*, 2002.
- [202] Mohamed Ben Salah, Amar Mitiche, and Ismail Ben Ayed. Effective level set image segmentation with a kernel induced data term. *IEEE Transactions on Image Processing*, 19(1):220–232, 2010.
- [203] Bogdan Savchynskyy. *Discrete Graphical Models - An Optimization Perspective*. under review, 2019.
- [204] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.
- [205] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.
- [206] Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.
- [207] Alexander G Schwing and Raquel Urtasun. Fully connected deep structured networks. *arXiv preprint arXiv:1503.02351*, 2015.
- [208] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, pages 524–535, 2018.
- [209] Laura Sevilla-Lara, Deqing Sun, Varun Jampani, and Michael J Black. Optical flow with semantic segmentation and localized layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3889–3898, 2016.

- [210] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
- [211] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.
- [212] Jamie Shotton, Toby Sharp, Pushmeet Kohli, Sebastian Nowozin, John Winn, and Antonio Criminisi. Decision jungles: Compact and rich models for classification. In *Advances in Neural Information Processing Systems (NIPS)*, pages 234–242, 2013.
- [213] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.
- [214] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. 2018.
- [215] K. K. Sung and T. Poggio. Example based learning for viewbased human face detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 20:39–51, 1995.
- [216] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- [217] Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Pseudo-bound optimization for binary energies. In *European Conference on Computer Vision (ECCV)*, pages 691–707, 2014.
- [218] Meng Tang, Ismail Ben Ayed, Dmitrii Marin, and Yuri Boykov. Secrets of grab-cut and kernel k-means. In *International Conference on Computer Vision (ICCV)*, Santiago, Chile, December 2015.
- [219] Meng Tang, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, and Christopher Schroers. Normalized Cut Loss for Weakly-supervised CNN Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [220] Meng Tang, Lena Gorelick, Olga Veksler, and Yuri Boykov. Grabcut in one cut. In *International Conference on Computer Vision (ICCV)*, Sydney, Australia, 2013.

- [221] Meng Tang, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. Kernel Cuts: MRF meets kernel and spectral clustering. In *arXiv:1506.07439*, Sept. 2016.
- [222] Meng Tang, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. Normalized Cut meets MRF. In *European Conference on Computer Vision (ECCV)*, Amsterdam, Netherlands, October 2016.
- [223] Meng Tang, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. Kernel Cuts: Kernel and spectral clustering meet regularization. *International Journal of Computer Vision (IJCV)*, 127:477–511, May 2019.
- [224] Meng Tang, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, and Yuri Boykov. On Regularized Losses for Weakly-supervised CNN Segmentation. In *European Conference on Computer Vision (ECCV)*, 2018.
- [225] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- [226] Ajanthan Thalaiyasingam, Alban Desmaison, Rudy Bunel, Mathieu Salzmann, Philip HS Torr, and M Pawan Kumar. Efficient linear programming for dense crfs. In *Conference on Computer Vision and Pattern Recognition*, 2017.
- [227] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.
- [228] Vladimir Vapnik. *Statistical Learning Theory*. Wiley, 1998.
- [229] Olga Veksler. Star shape prior for graph-cut image segmentation. In *European Conference on Computer Vision*, pages 454–467. Springer, 2008.
- [230] Olga Veksler. Multi-label moves for mrfs with truncated convex priors. *International journal of computer vision*, 98(1):1–14, 2012.
- [231] Olga Veksler. Efficient graph cut optimization for full crfs with quantized edges. *IEEE transactions on pattern analysis and machine intelligence*, 2019.
- [232] Paul Vernaza and Manmohan Chandraker. Learning random-walk label propagation for weakly-supervised semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2017.

- [233] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Graph cut based image segmentation with connectivity priors. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
- [234] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Joint optimization of segmentation and appearance models. In *International Conf. on Computer Vision (ICCV)*, 2009.
- [235] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.
- [236] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE transactions on information theory*, 51(11):3697–3717, 2005.
- [237] Naiyan Wang and Dit-Yan Yeung. Learning a deep compact image representation for visual tracking. In *Advances in neural information processing systems*, pages 809–817, 2013.
- [238] Song Wang and Jeffrey Mark Siskind. Image segmentation with ratio cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(6):675–690, 2003.
- [239] Yair Weiss and William T Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, 47(2):736–744, 2001.
- [240] Tomáš Werner. A linear programming approach to max-sum problem: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(7):1165–1179, 2007.
- [241] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.
- [242] Oliver Woodford, Philip Torr, Ian Reid, and Andrew Fitzgibbon. Global stereo reconstruction under second-order smoothness priors. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2115–2128, 2009.
- [243] Oliver J. Woodford, Carsten Rother, and Vladimir Kolmogorov. A global perspective on map inference for low-level vision. In *ICCV*, pages 2319–2326, 2009.

- [244] Jia Xu, Alexander G Schwing, and Raquel Urtasun. Learning to segment under various forms of weak supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3781–3790, 2015.
- [245] Linli Xu, Wenye Li, and Dale Schuurmans. Fast normalized cut with linear constraints. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2866–2873, 2009.
- [246] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–381, 2016.
- [247] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239, 2003.
- [248] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005.
- [249] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659, 2016.
- [250] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.
- [251] Stella Yu and Jianbo Shi. Multiclass spectral clustering. In *International Conference on Computer Vision (ICCV)*, 2003.
- [252] Stella X Yu and Jianbo Shi. Segmentation given partial grouping constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(2):173–183, 2004.
- [253] Yizhou Yu, Chaowei Fang, and Zicheng Liao. Piecewise flat embedding for image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1368–1376, 2015.
- [254] Jing Yuan, Egil Bae, and Xue-Cheng Tai. A study on continuous max-flow and min-cut approaches. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2217–2224. IEEE, 2010.

- [255] Y. Yuan. A review of trust region algorithms for optimization. In *Proceedings of the Fourth International Congress on Industrial and Applied Mathematics (ICIAM)*, 1999.
- [256] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in NIPS*, pages 1601–1608, 2004.
- [257] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Amrith Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [258] Zhihua Zhang, James T. Kwok, and Dit-Yan Yeung. Surrogate maximization/minimization algorithms and extensions. *Machine Learning*, 69:1–33, 2007.
- [259] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017.
- [260] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015.
- [261] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.
- [262] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, pages 321–328, 2004.
- [263] Song Chun Zhu and Alan Yuille. Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(9):884–900, Sept. 1996.
- [264] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.
- [265] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

APPENDICES

Appendix A

Matlab Code for Kernel and Spectral Bound ¹

A.1 Kernel Bound (3.52)

```
function [ unaries ] = KernelBound( A, K, current_clustering, energy_type)
%KERNELBOUND derives linear upper bound w.r.t binary indicator variables
%for NC (Normalized Cut) or AA (Average Association)
%Inputs:
%  A --- Affinity matrix (sparse or full) of N-by-N
%  K --- Number of desired clusters, default is 2
%  current_clustering --- N-by-1 vector of cluster indicator, value from 1
%  to K, default is a random generated vector
%  energy_type --- String 'NC' or 'AA', default is 'NC'
%Outputs:
%  unaries --- N-by-K matrix where unaries(n,k) is the unary (linear) cost
%  of assigning data n to cluster k

N = size(A,1); % number of data points
if nargin < 2; K = 2; end
if nargin < 3; current_clustering = randi([1, K], N, 1); end
if nargin < 4; energy_type = 'NC'; end
```

¹Also in <https://github.com/meng-tang/KernelCut/blob/master/matlab>

```

if ~strcmp(energy_type, 'NC') && ~strcmp(energy_type, 'AA')
    error('Energy type has to be NC (normalized cut) or AA (average association).')
end

unaries = zeros(N, K, 'double');
% degree vector
d = sum(A,2);
for i=1:K
    % current binary indicators (N-by-1) for cluster i
    S_t = double(current_clustering == i);
    % compute gradient as unaries (see kernel bound proposition in the paper)
    if strcmp(energy_type, 'NC') % for normalized cut
        unaries(:,i) = S_t'* A * S_t / (d' * S_t)^2 * d - 2 * A * S_t / (d' * S_t);
    elseif strcmp(energy_type, 'AA') % for average association
        unaries(:,i) = S_t'* A * S_t / (ones(1,N) * S_t)^2 * ones(N,1) - 2 * A * S_t /
    end
end
end
end

```

A.2 Spectral Bound (3.66)

```

function [ unaries ] = SpectralBound( A, K, m, current_clustering, energy_type)
%KERNELBOUND derives linear upper bound w.r.t binary indicator variables
%for NC (Normalized Cut) or AA (Average Association). This function first
%finds spectral embedding (for once) and take distances to (weighted) mean
%on such embeddings as unaries.
%Inputs:
%  A --- Affinity matrix (sparse or full) of N-by-N
%  K --- Number of desired clusters, default is 2
%  m --- dimensionality of embedding, typically small
%  current_clustering --- N-by-1 vector of cluster indicator, value from 1
%  to K, default is a random generated vector
%  energy_type --- String 'NC' or 'AA', default is 'NC'
%Outputs:
%  unaries --- N-by-K matrix where unaries(n,k) is the unary (linear) cost
%  of assigning data n to cluster k

```

```

N = size(A,1); % number of data points
if nargin < 2; K = 2; end
if nargin < 3; current_clustering = randi([1, K], N, 1); end
if nargin < 4; energy_type = 'NC'; end

if ~strcmp(energy_type, 'NC') && ~strcmp(energy_type, 'AA')
    error('Energy type has to be NC (normalized cut) or AA (average association).')
end

% for weighted k-means, only need to be computed ONCE.
persistent embedding weights
if isempty(embedding) || isempty(weights)
    [ embedding, weights ] = SpectralEmbedding( A, m, energy_type);
end

% update cluster centers for (weighted k-means)
centers = zeros(K,m);
weightedembedding = embedding .* repmat(weights,1,m);
for j=1:K
    centers(j,:) = sum( weightedembedding(current_clustering==j,:),1 ) ...
        / sum( weights(current_clustering==j));
end
% distances to centers
dittocenters = zeros(N,K);
for j=1:K
    dittocenters(:,j) = sum( (embedding - repmat(centers(j,:),N,1) ).^2 , 2);
end
% unaries are just distances to centers
unaries = dittocenters;

end

function [ embedding, weights ] = SpectralEmbedding( A, m, energy_type)
%SPECTRALEMMBEDDING embeds graph to m-dimension space such that (weighted)
%k-means on such embedding approximates NC (Normalized Cut) or AA (Average
%Association)
%Inputs:

```

```

% A --- Affinity matrix (sparse or full) of N-by-N
% m --- dimensionality of embedding, typically small
% energy_type --- String 'NC' or 'AA', default is 'NC'
%Outputs:
% embedding --- N-by-m matrix n th row is the embedding for n th node
% weights --- N-by-1 vector denoting weights of each embedding

N = size(A,1); % number of data points
if nargin < 2; m = 2; end
if nargin < 3; energy_type = 'NC'; end
if ~strcmp(energy_type, 'NC') && ~strcmp(energy_type, 'AA')
    error('Energy type has to be NC (normalized cut) or AA (average association).')
end

opts.issym=1;
opts.isreal = 1;
opts.disp=0;
if strcmp(energy_type, 'AA') % for average association
    tic
    [EigVect, EVal] = eigs(A, m, 'lm',opts);
    embedding = EigVect * sqrt(EVal);
    weights = ones(N,1);
    toc
    return;
elseif strcmp(energy_type, 'NC') % for normalized cut
    % degree vector
    d = sum(A,2);
    % degree matrix
    D = sparse(1:N,1:N,d);
    if ~issparse(A); D = full(D); end;
    tic
    [EigVect, EVal] = eigs(D^(-0.5)*A*D^(-0.5), m, 'lm',opts);
    toc
    embedding = EigVect * sqrt(EVal)./repmat(sqrt(d),1,m);
    weights = d;
    return
end
end

```


Appendix B

PyTorch Code for Regularized Loss ¹

B.1 DenseCRF Loss

```
import torch
import torch.nn as nn
from torch.autograd import Function
from torch.autograd import Variable
import torch.nn.functional as F
import numpy as np
import sys
sys.path.append("../wrapper/bilateralfilter/build/lib.linux-x86_64-3.6")
from bilateralfilter import bilateralfilter, bilateralfilter_batch
from dataloaders.custom_transforms import denormalizeimage
import time
from multiprocessing import Pool
import multiprocessing
from itertools import repeat
import pickle
```

```
class DenseCRFLossFunction(Function):
```

```
    @staticmethod
```

¹Also in <https://github.com/meng-tang/rloss/tree/master/pytorch>

```

def forward(ctx, images, segmentations, sigma_rgb, sigma_xy, ROIs):
    ctx.save_for_backward(segmentations)
    ctx.N, ctx.K, ctx.H, ctx.W = segmentations.shape

    ROIs = ROIs.unsqueeze_(1).repeat(1, ctx.K, 1, 1)
    segmentations = torch.mul(segmentations.cuda(), ROIs.cuda())
    ctx.ROIs = ROIs

    densecrf_loss = 0.0
    images = images.numpy().flatten()
    segmentations = segmentations.cpu().numpy().flatten()
    AS = np.zeros(segmentations.shape, dtype=np.float32)
    bilateralfilter_batch(images, segmentations, AS,
        ctx.N, ctx.K, ctx.H, ctx.W, sigma_rgb, sigma_xy)
    densecrf_loss -= np.dot(segmentations, AS)

    # averaged by the number of images
    densecrf_loss /= ctx.N

    ctx.AS = np.reshape(AS, (ctx.N, ctx.K, ctx.H, ctx.W))
    return Variable(torch.tensor([densecrf_loss]), requires_grad=True)

    @staticmethod
    def backward(ctx, grad_output):
        grad_segmentation = -2*grad_output*torch.from_numpy(ctx.AS)/ctx.N
        grad_segmentation=grad_segmentation.cuda()
        grad_segmentation = torch.mul(grad_segmentation, ctx.ROIs.cuda())
        return None, grad_segmentation, None, None, None

class DenseCRFLoss(nn.Module):
    def __init__(self, weight, sigma_rgb, sigma_xy, scale_factor):
        super(DenseCRFLoss, self).__init__()
        self.weight = weight
        self.sigma_rgb = sigma_rgb
        self.sigma_xy = sigma_xy
        self.scale_factor = scale_factor

```

```

def forward(self, imgs, segmentations, ROIs):
    """ scale imag by scale_factor """
    scaled_images = F.interpolate(imgs, scale_factor=self.scale_factor)
    scaled_segs = F.interpolate(segmentations,
                                scale_factor=self.scale_factor,
                                mode='bilinear',
                                align_corners=False)
    scaled_ROIs = F.interpolate(ROIs.unsqueeze(1),
                                scale_factor=self.scale_factor)
                                .squeeze(1)
    return self.weight*DenseCRFLossFunction.apply(
        scaled_images,
        scaled_segs,
        self.sigma_rgb,
        self.sigma_xy*self.scale_factor,
        scaled_ROIs)

def extra_repr(self):
    return 'sigma_rgb={},_sigma_xy={},_weight={},_scale_factor={}'
    .format(self.sigma_rgb, self.sigma_xy,
            self.weight, self.scale_factor)

```